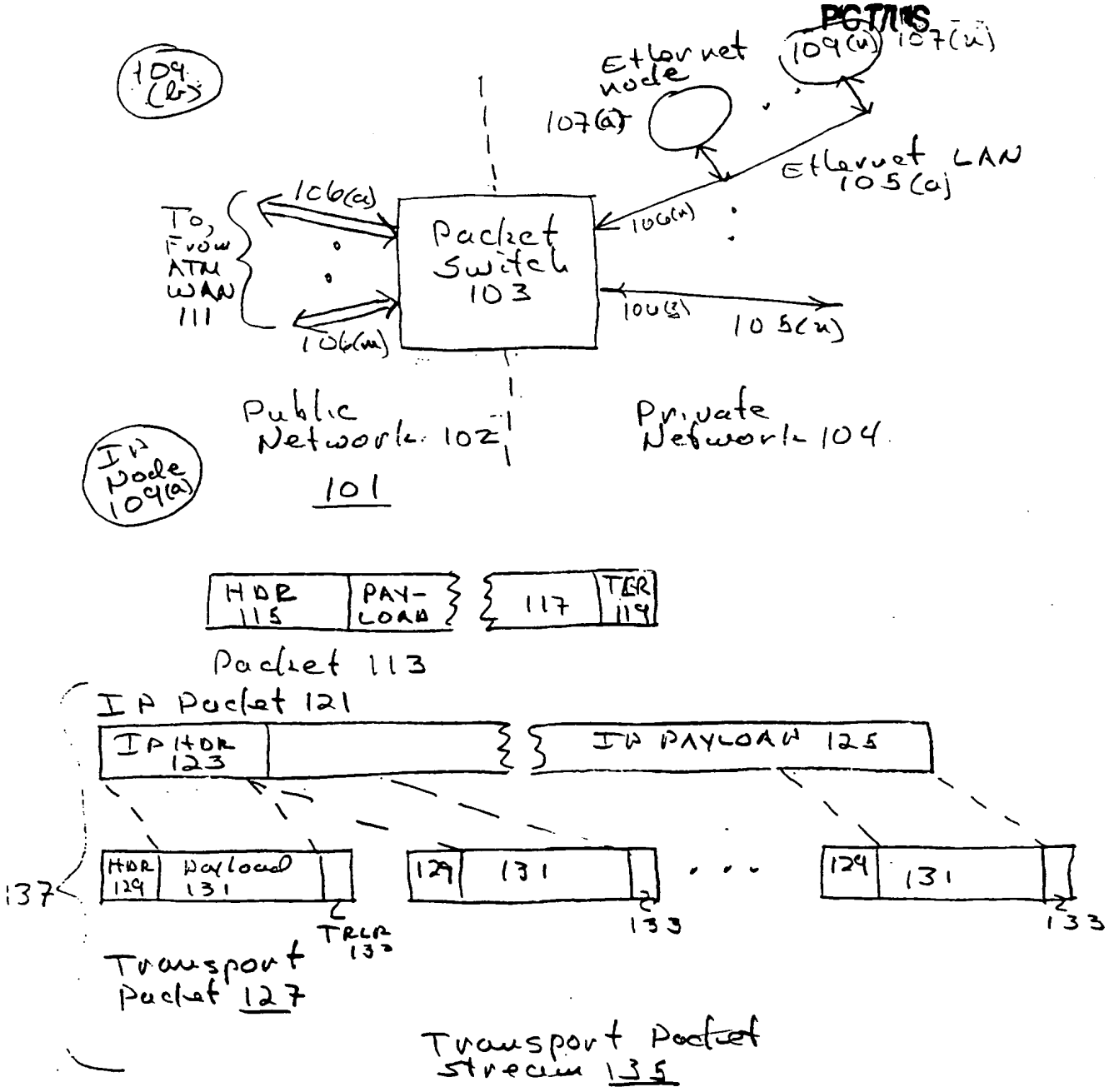
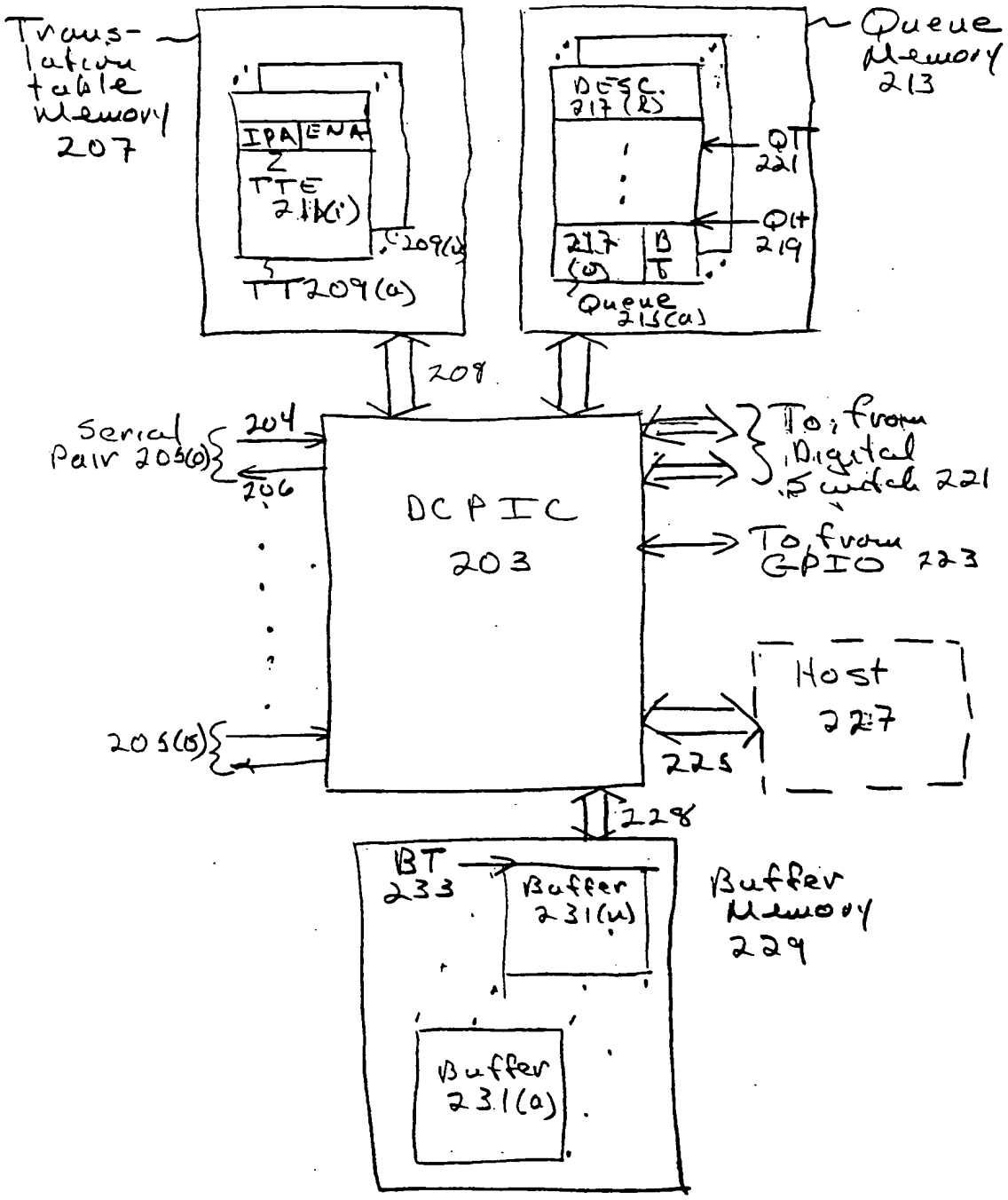


Fig. 1 Prior Art


Trans-
lation
table
Memory
207
IPA
ENA
Z
TTE
211(i)
209(i)
TT209(a)
DESC.
217(k)
QT
221
QH
219
217
(a)
B
T
Queue
215(a)
Queue
Memory
213
209
Serial
Pair 205(0)
204
206
DCPIC
203
To, from
Digital
Switch 221
To, from
GPIO 223
Host
227
225
205(5)
228
BT
233
Buffer
231(u)
Buffer
231(a)
Buffer
Memory
229

201

Fig. 2

Figure 1-DCP-1 Block Diagram

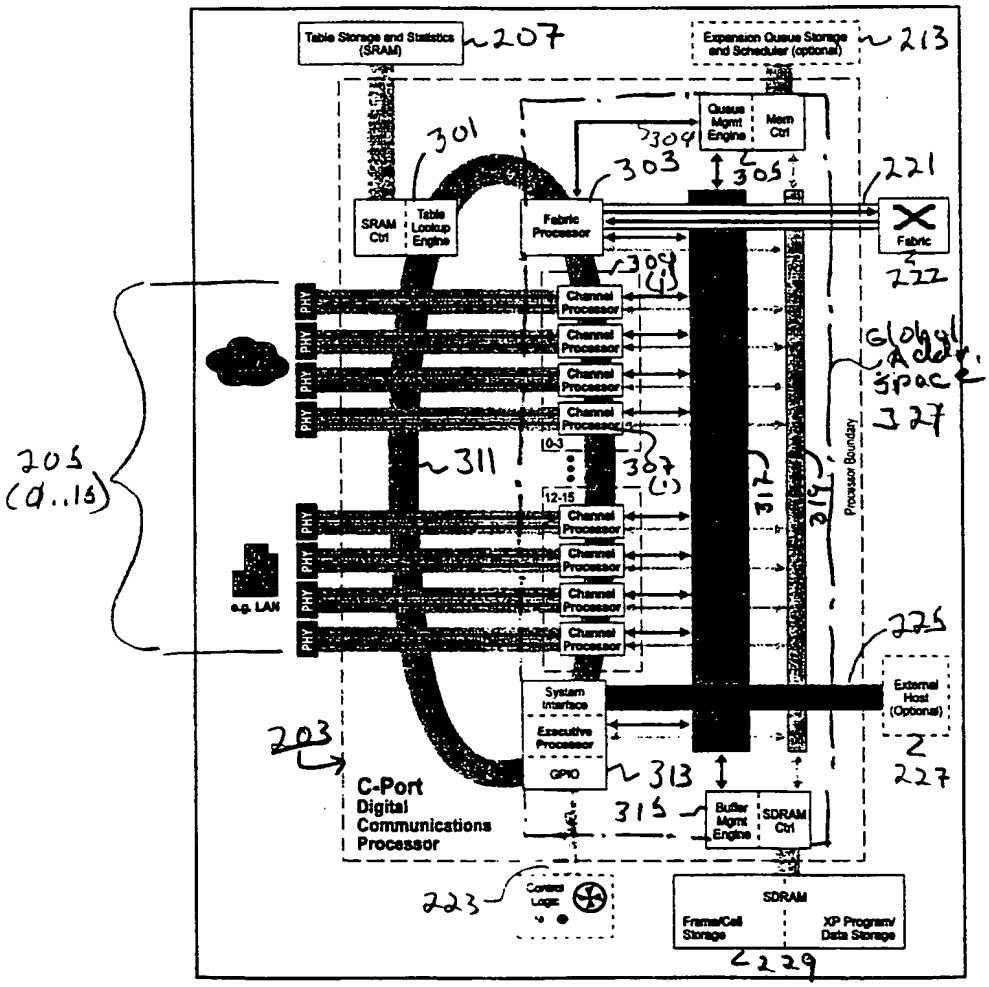


Fig. 3

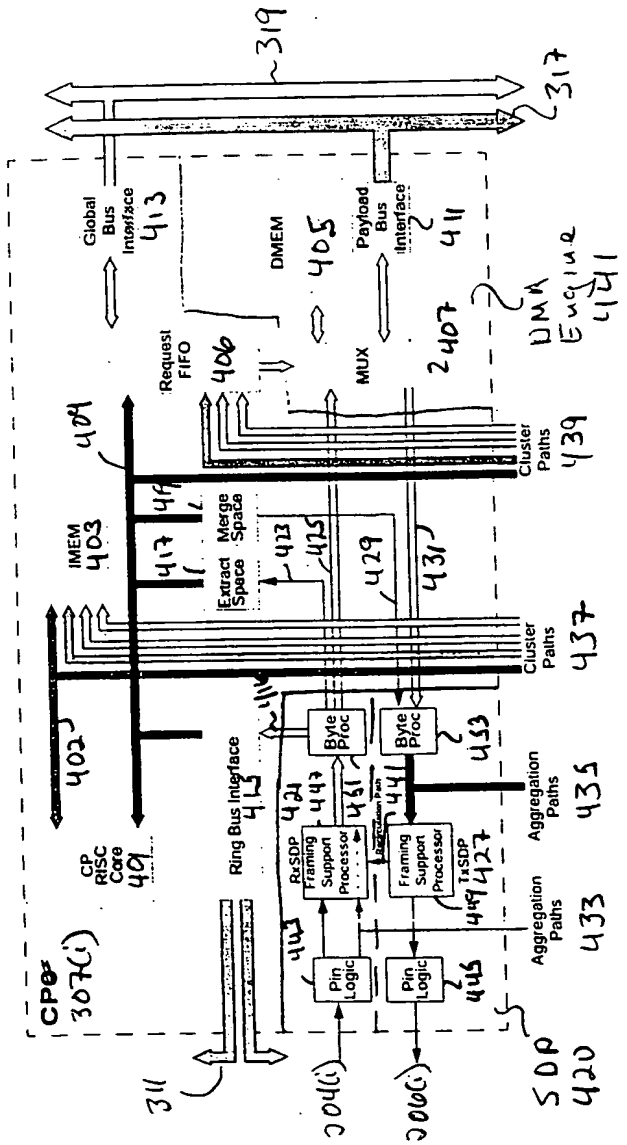

319
317
Global Bus Interface
413
DMEM
405
Payload Bus Interface
411
Request FIFO
406
MUX
407
DMA Engine
441
409
IMEM
403
Extract Space
Merge Space
413
415
417
429
431
Cluster Paths
439
Cluster Paths
437
402
Ring Bus Interface
415
CP RISC Core
401
Byte Proc
Byte Proc
453
Aggregation Paths
435
RxSDP Framing Support Processor
TxSDP Framing Support Processor
449
427
Aggregation Paths
433
Pin Logic
Pin Logic
445
CPs
307(i)
311
SDP
420

Fig. 4

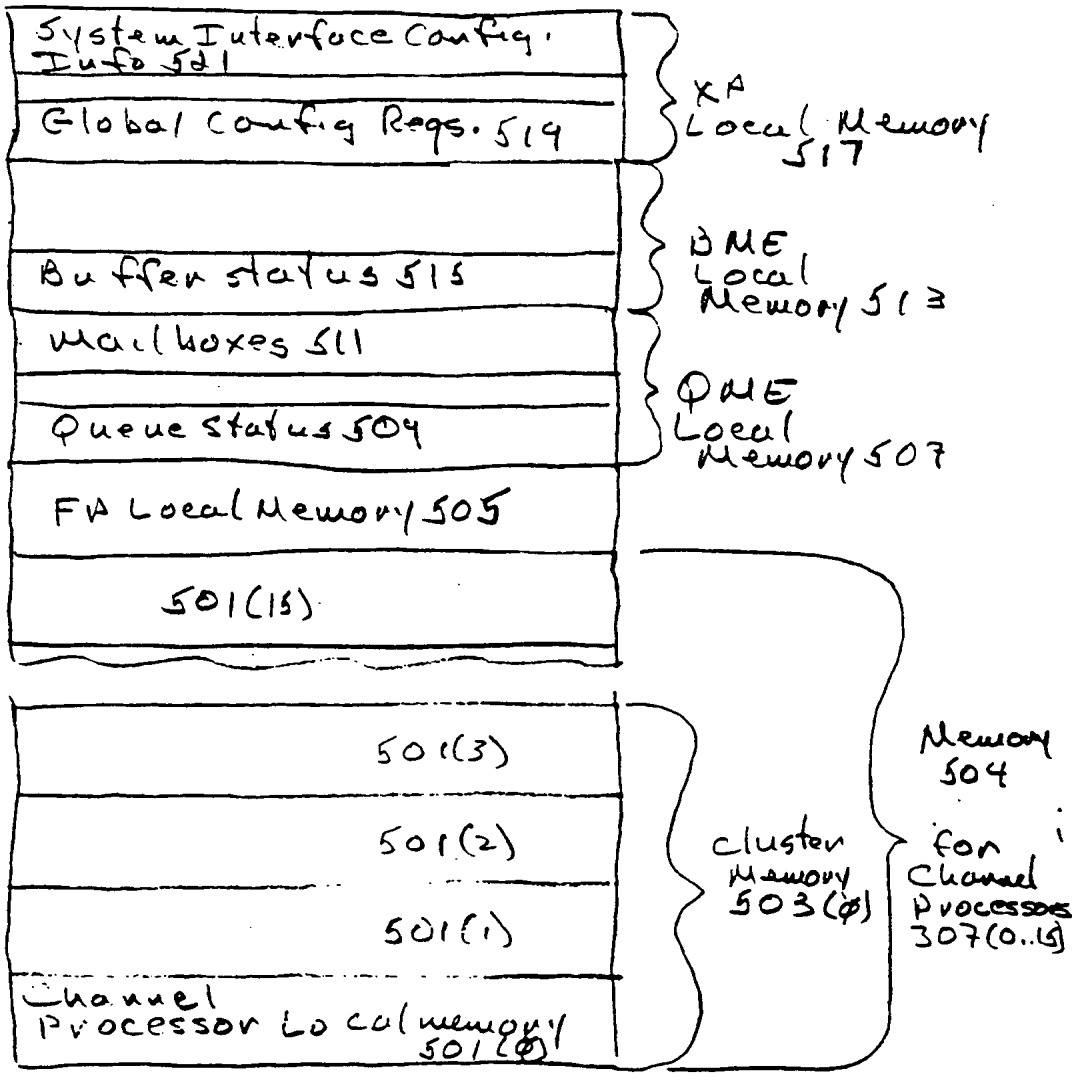


Fig. 5

09/674864

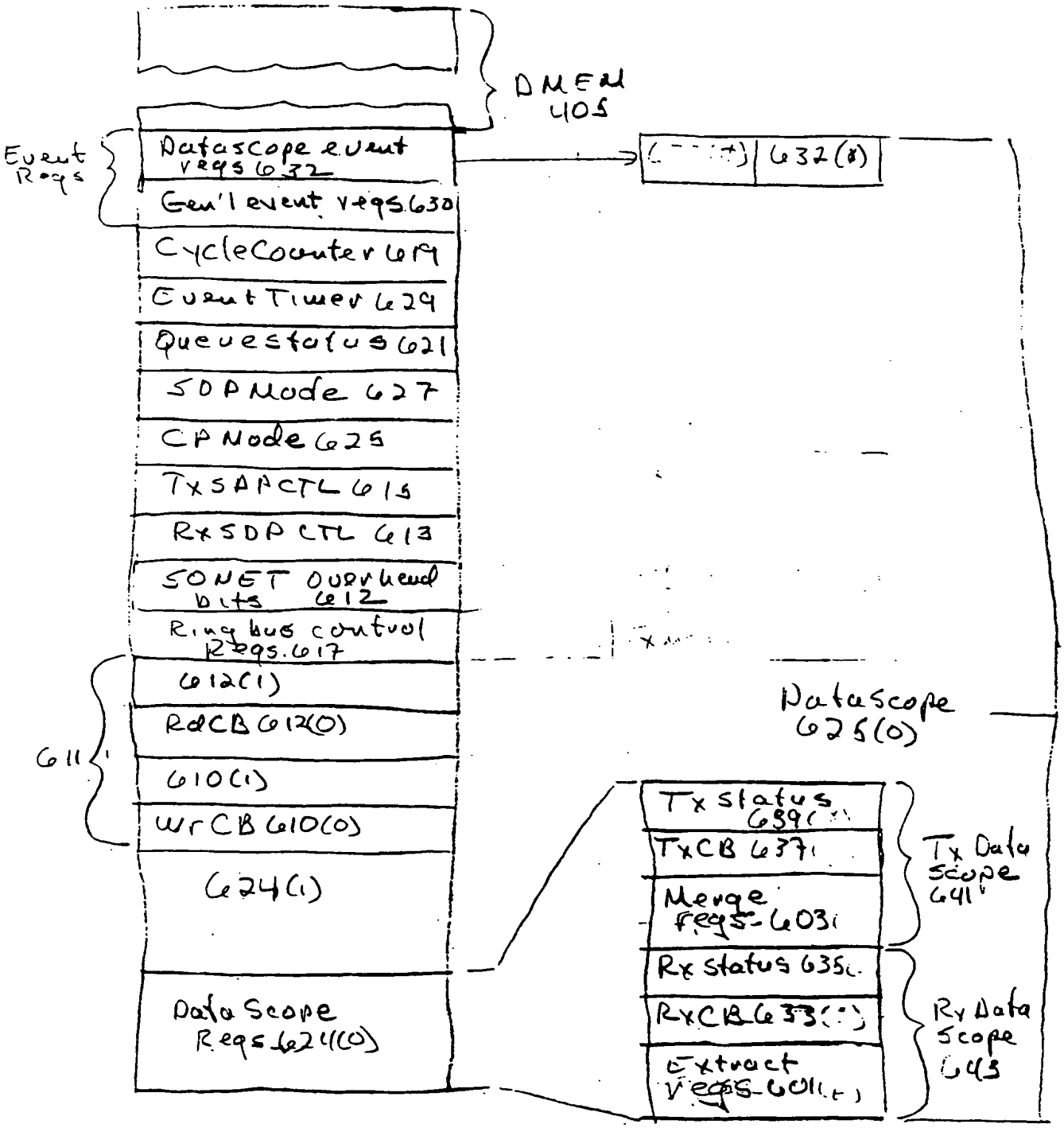
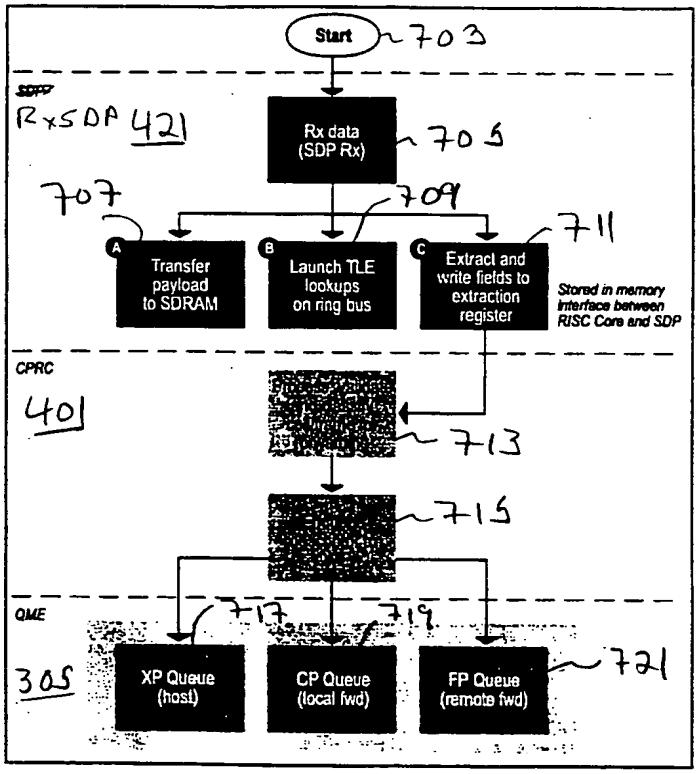


Fig. 6

Figure 2 DCP-1 Receive Thread of Execution Flowchart



701

Fig 7

09/674864

Figure 3 DCP-1 Transmit Execution Thread Flowchart

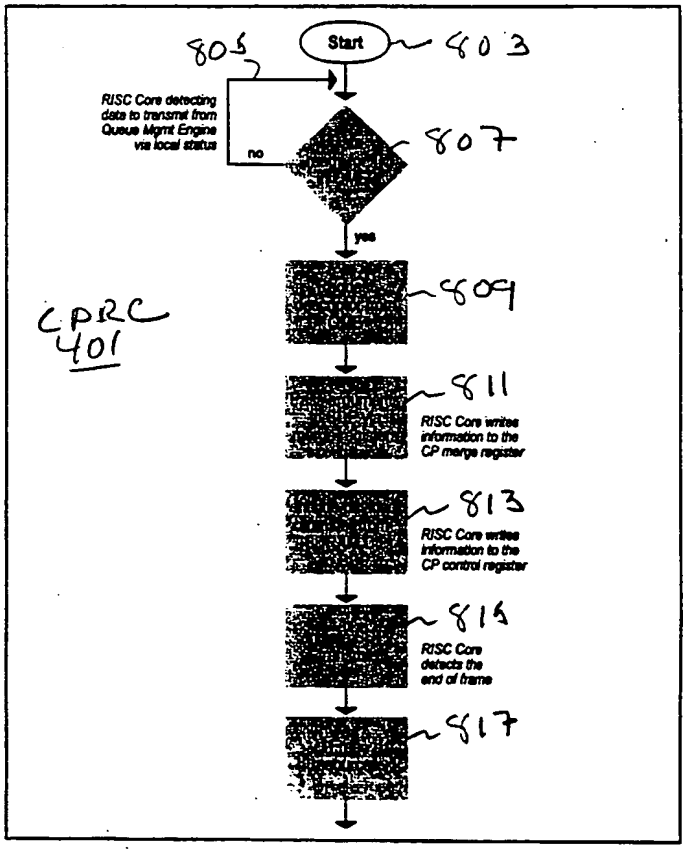


Fig. 4

OWN 935	LS:LØ 937	Busy 941

635

BTAG 933			
OFFSET 931			
AU 929	NR 927	EVV 925	OWN 921
SOPST 915	EOP 927	BCTL ST 919	
Length 911			
Buffer Pool No 909			
DMEM ONRAMADR 907			
Tx Rcy Addr. 905			
Rx Rcy Addr. 903			
DMEM Byte Addr 901			

RxCBCTL 913

633

Fig. 9

09/674864

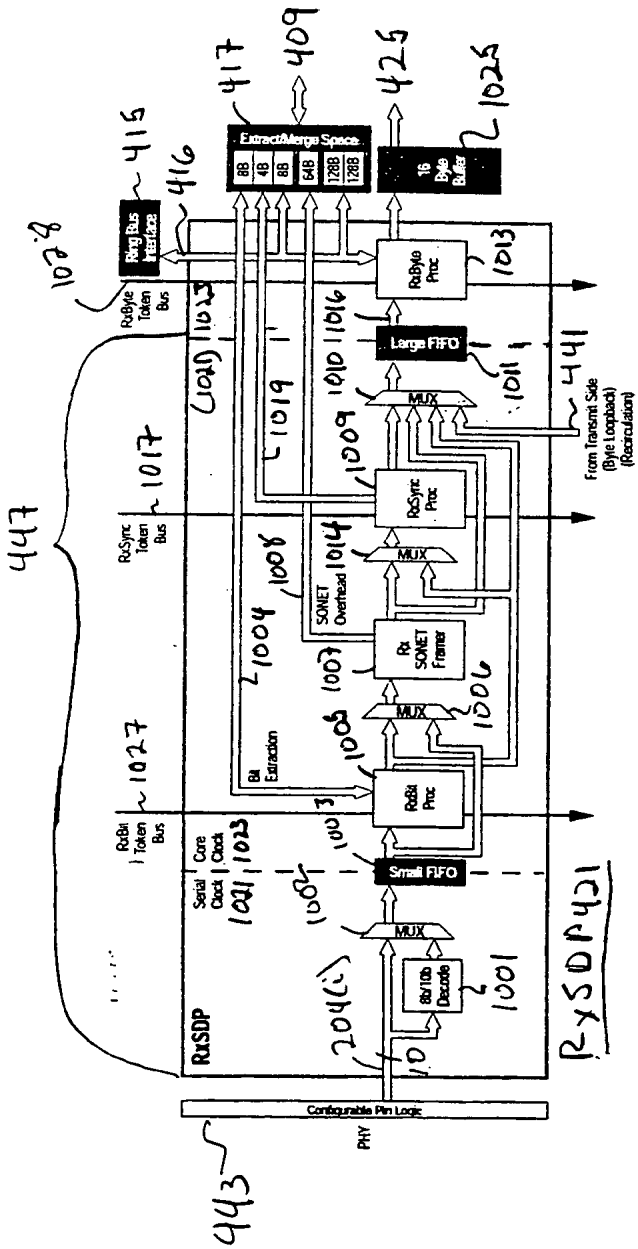
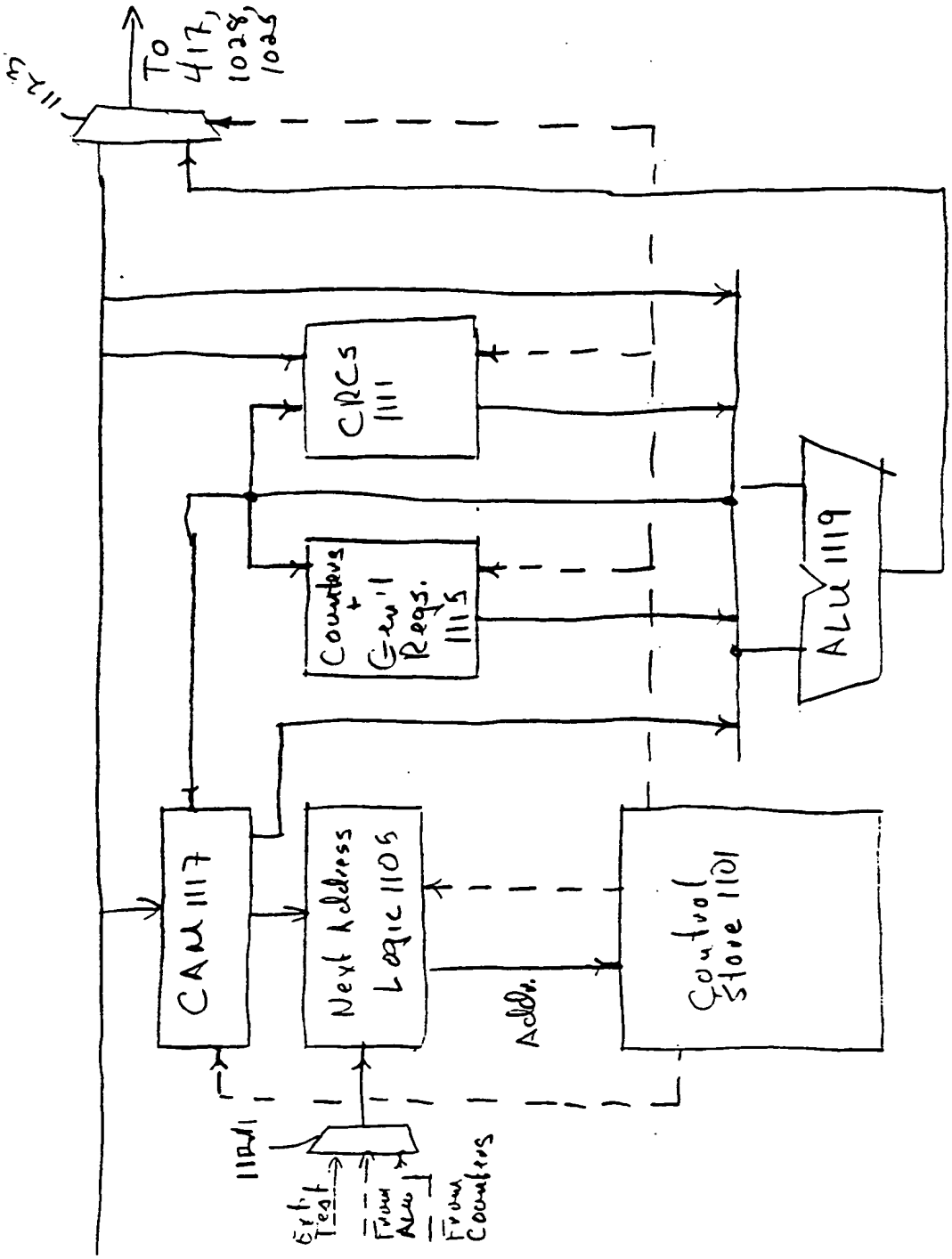


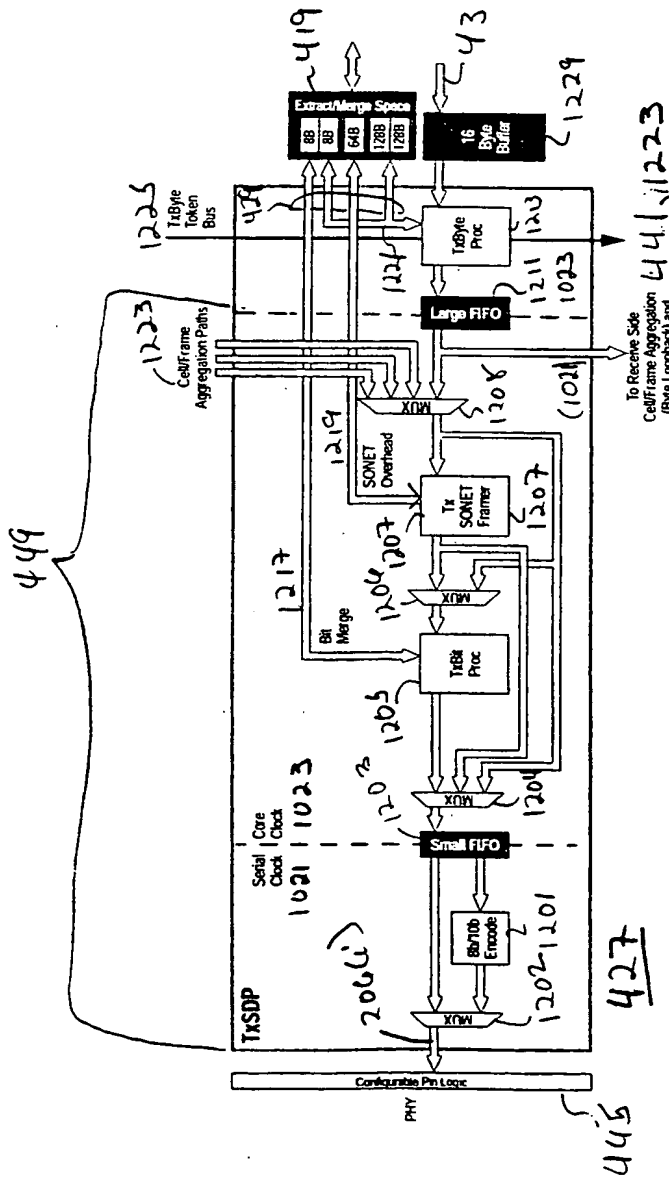
Fig. 10

09/674864



1013

Fig. 11


Extract/Merge Space
8B
8B
64B
128B
128B
16 Byte Buffer
TxByte Token Bus
TxByte Proc
Large FIFO
Cell/Frame Aggregation Paths
To Receive Side Cell/Frame Aggregation (Byte Loopback) and Tx Aggregation
MUX
SONET Overhead
Tx SONET Framer
Bit Merge
TxBit Proc
Core Clock
Serial Clock
Small FIFO
8b/10b Encode
TxSDP
Configurable Pin Logic
PHY

09/674864

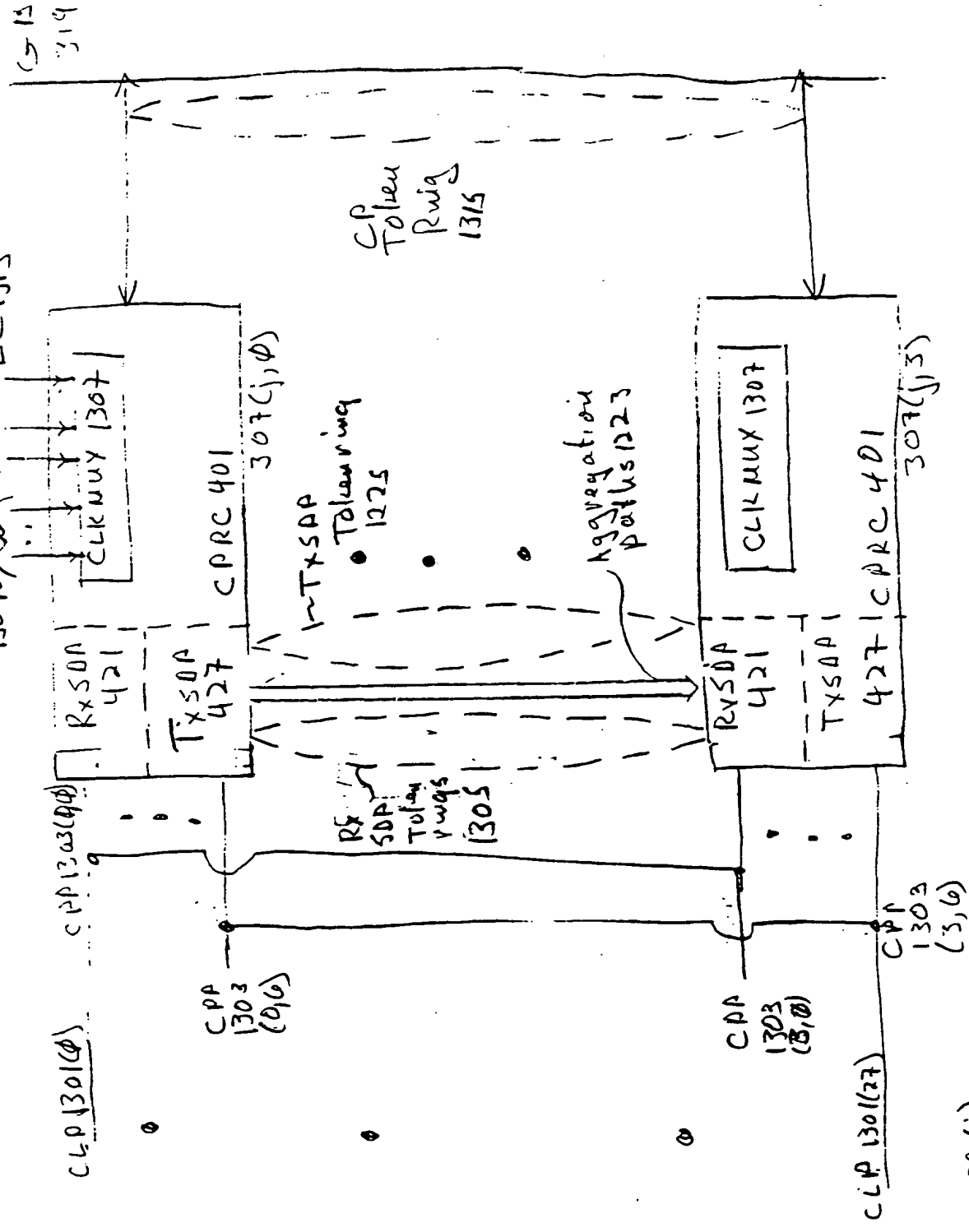


Fig. 13

09/674864

CP Instruction Memory

Each channel shares access to a 16kB IMEM among a cluster of four adjacent CPs as shown in Figure 7. The IMEM is configured as four sub-arrays, with each CP in the cluster given access to the arrays, one per cycle, in fixed round-robin order. With this simple interleaved scheme, the four adjacent RCs can access this memory at nearly full bandwidth.

When adjacent channels are configured to handle similar communication protocols, the large shared memory can contain both channel-specific code and cluster-shared code such as exception routines.

At initialization time, the 16kB array can be divided so that each CP gets a dedicated 4kB sub-array. This array allocation removes all CP contention for IMEM (but also removes the opportunity to share code among CPs). The memory configuration options result in roughly the CPRC performance shown in Table 6-1 for non-blocking code. The *optimized* column means that the compiler has placed code such that the branch target address bits <3:2> equal the branch fall through address bits <3:2>. CPRC instruction references outside of the shared local memory space are not supported.

Table 3 Channel RISC Core Instruction Execution Efficiency

IPC	IPC optimized	CP IMEM configuration
.85	.90	4 CPs sharing 16kB
.95	N/A	each CP accessing a single 4kB sub-array

~~Figure 7 Local and Shared Memory in a Channel Cluster~~

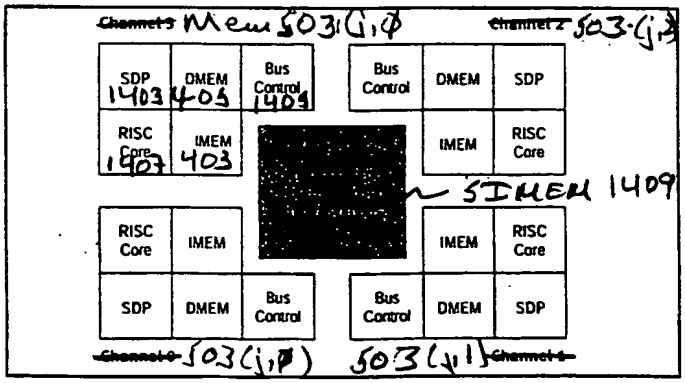


Fig. 14

509

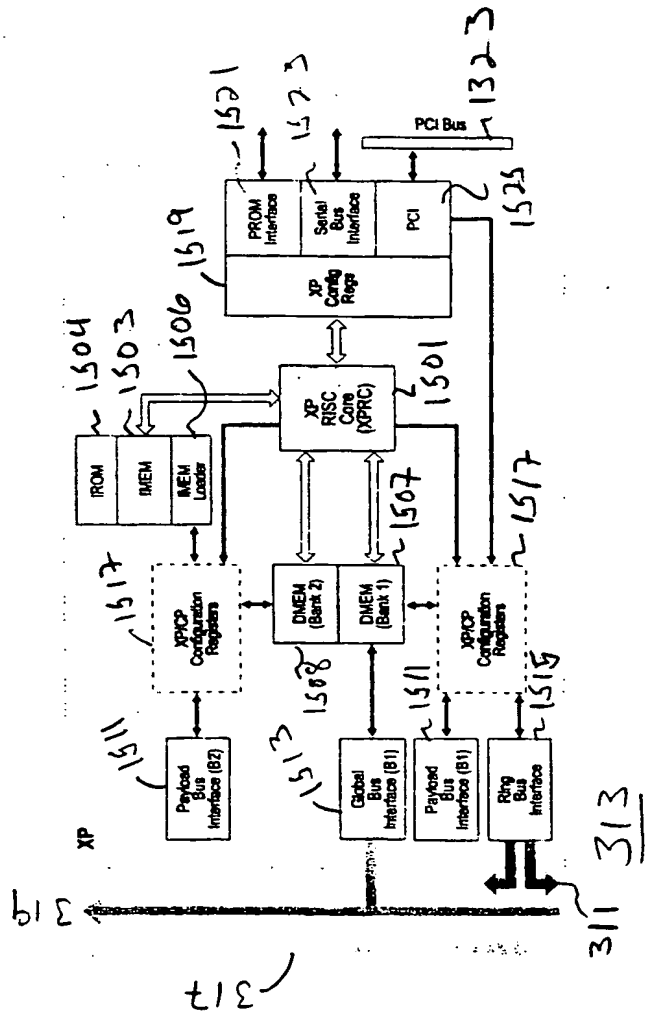

PCI Bus
PROM Interface
Serial Bus Interface
PCI
XP Config Regs
XP RISC Core (XPRC)
IROM
IMEM
IMEM Loader
XP/CP Configuration Registers
DMEM (Bank 2)
DMEM (Bank 1)
XP/CP Configuration Registers
Payload Bus Interface (B2)
Global Bus Interface (B1)
Payload Bus Interface (B1)
Ring Bus Interface
XP
1521
1523
1525
1519
1504
1503
1506
1501
1507
1517
1512
1508
1511
1513
1515
313
311
319
317

Fig. 15

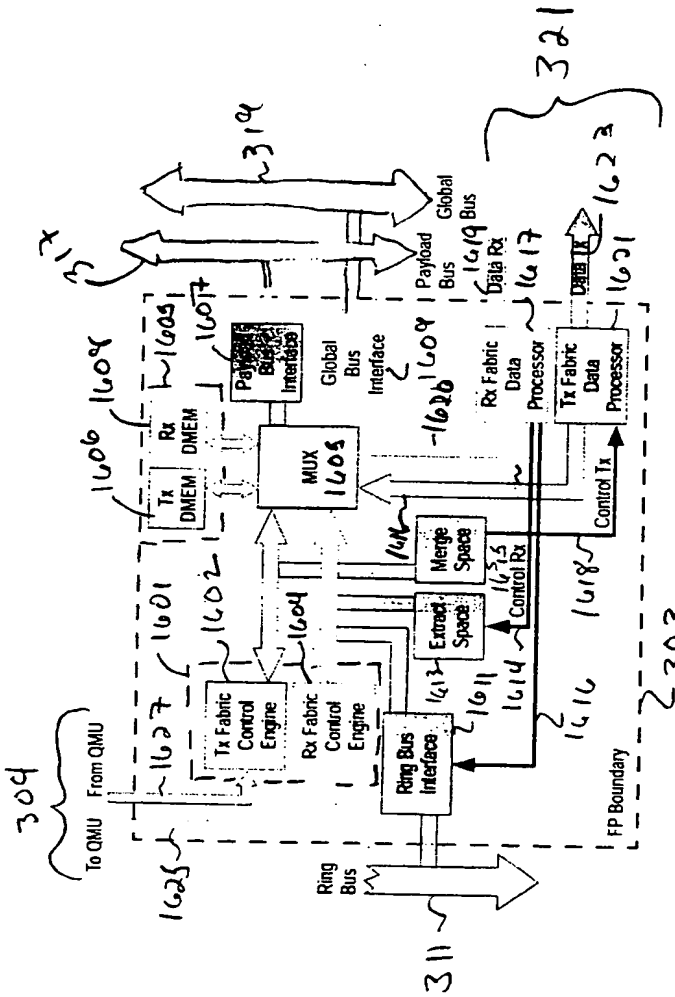

304
To QMU
From QMU
1625
1627
1601
1602
1604
Tx Fabric Control Engine
Rx Fabric Control Engine
Ring Bus Interface
Ring Bus
311
1611
1613
Extract Space
1614
1615
Control Rx
Merge Space
1618
1616
FP Boundary
303
MUX 1605
1606
Tx DMEM
Rx DMEM
1607
Payload Bus Interface
1609
Global Bus Interface
1620
Rx Fabric Data Processor
Tx Fabric Data Processor
Control Tx
1617
Data Rx
1619
Payload Bus
Global Bus
1621
Data Tx
1623
321
314
312

Fig. 16

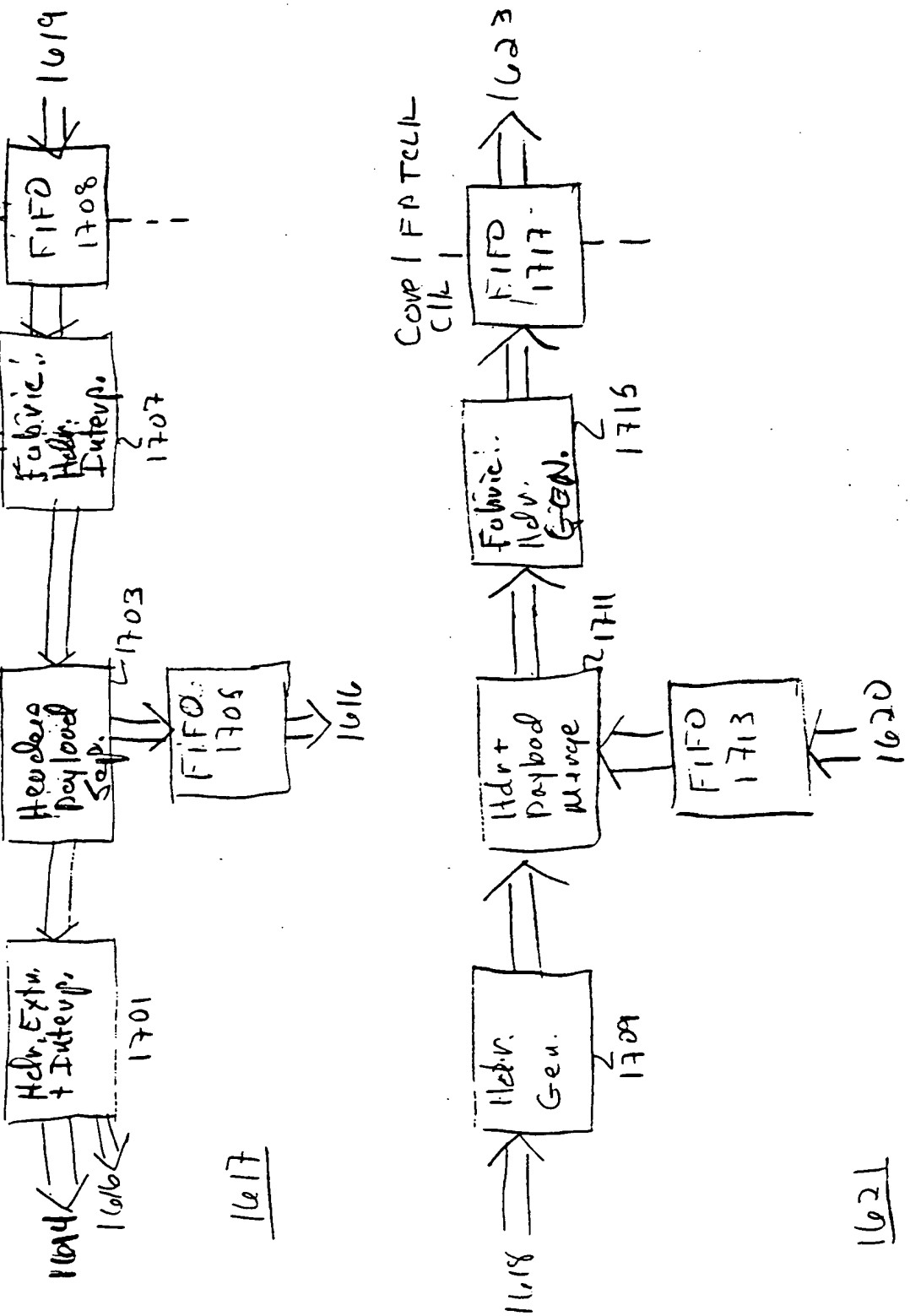

Core Fabric Clk
1619
FIFO 1708
1614
Fabric Hdr. Interp.
1707
1703
Header Payload Sep.
FIFO 1705
1616
Hdr. Extr. + Interp.
1701
1604
1606
1617
Core Fabric Clk
1623
FIFO 1717
Fabric Hdr. Gen.
1715
1711
Hdr + Payload Merge
FIFO 1713
1620
Hdr. Gen.
1709
1618
1621

Fig. 17

09/674864

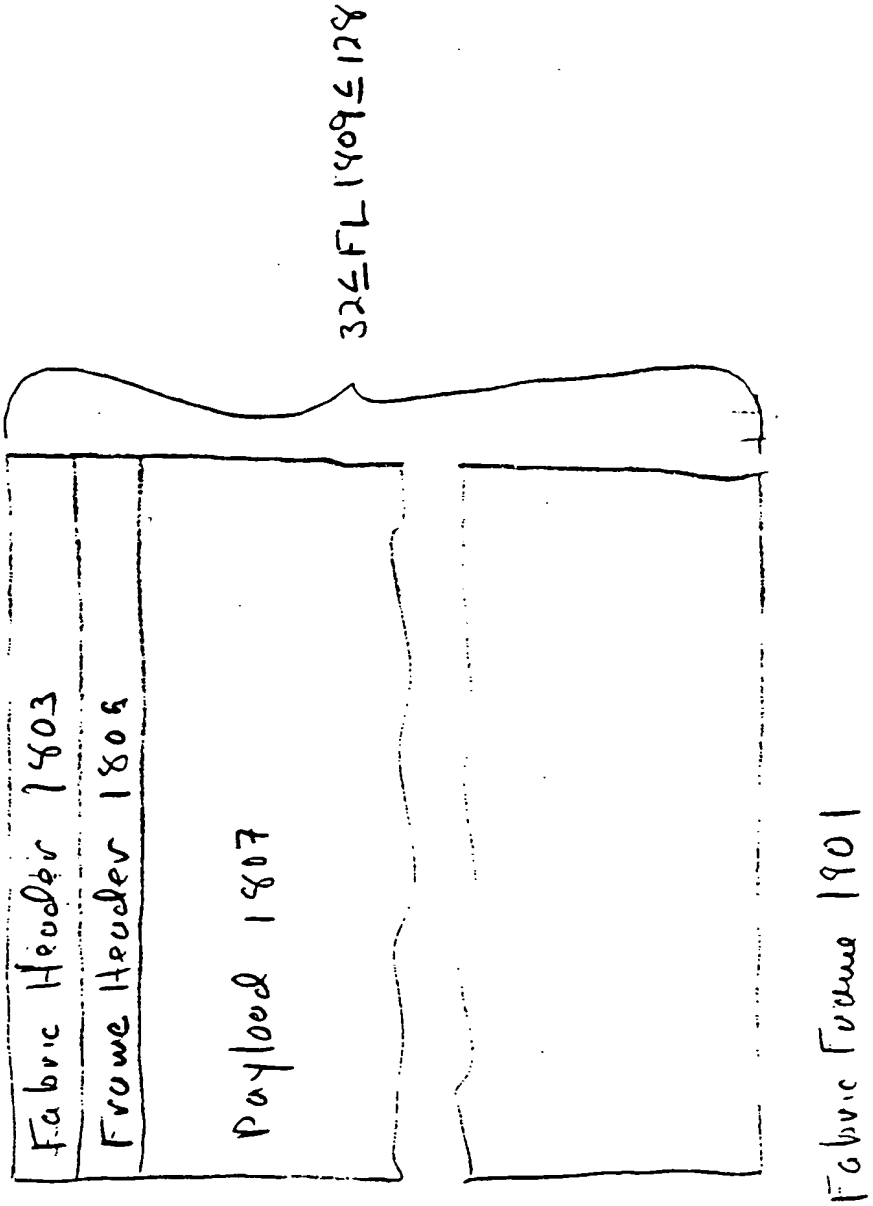
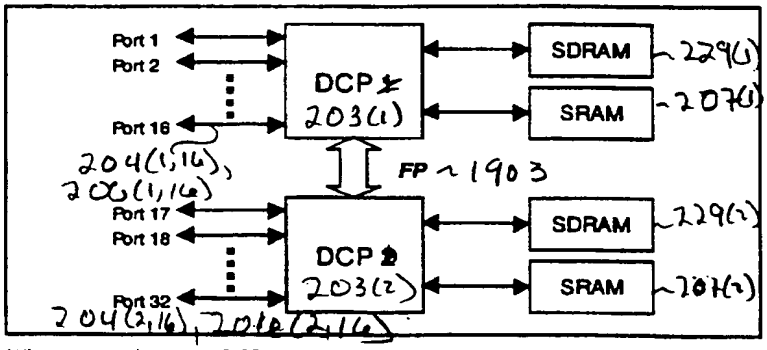


Fig. 18

09/674864

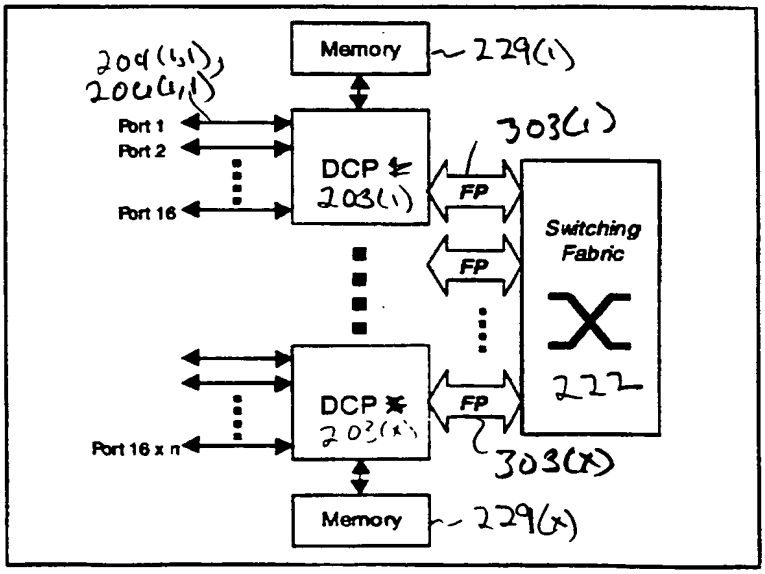
Figure 12 Two DCP-1 Application



1905

When more than two DCP-1s are required in a system, a switching fabric is utilized. The switching solution has two or more FP-type ports and provides a mechanism for switching cell- or packet-based data from one DCP-1 to another. An homogenous, multi-DCP-1 application is shown in Figure 13.

Figure 13 Multiple DCPs with Switching Fabric



1901

Fig. 19

09/674868

Figure 14 Heterogeneous DCP-1 Switching Application

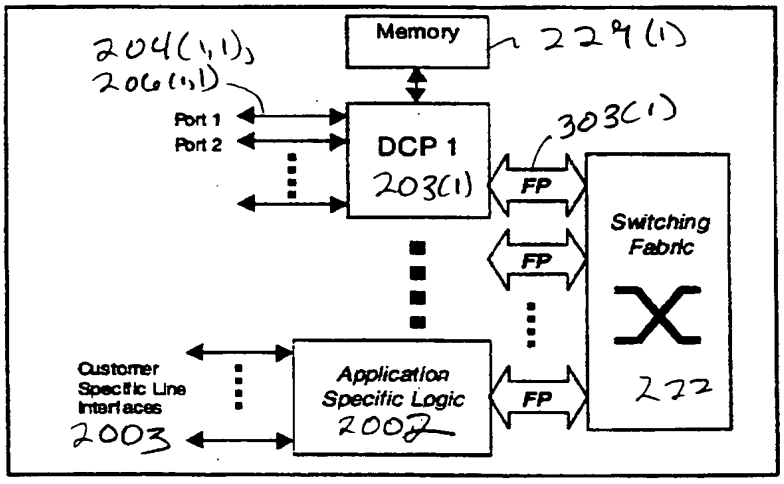


Fig. 20

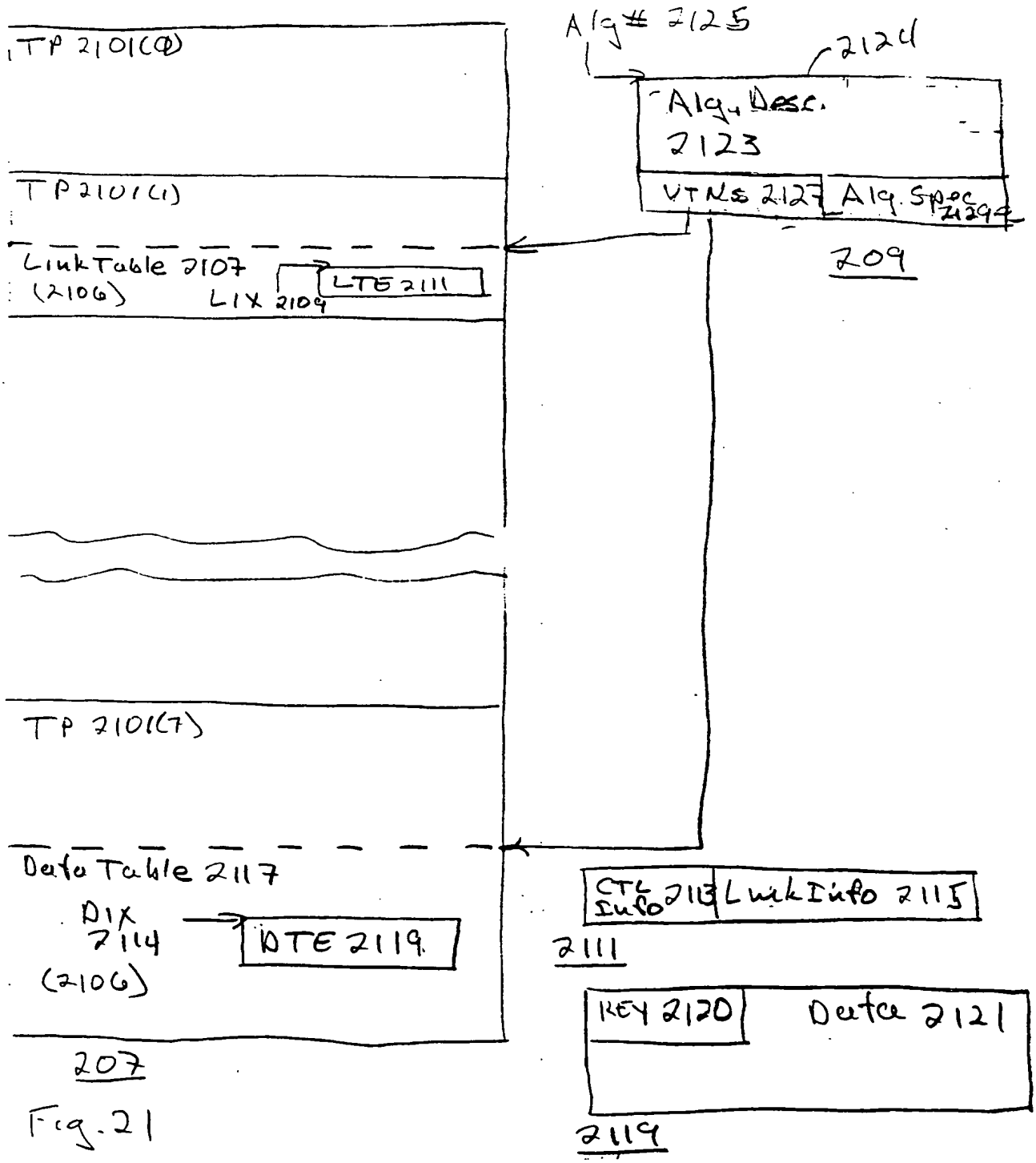


Fig. 21

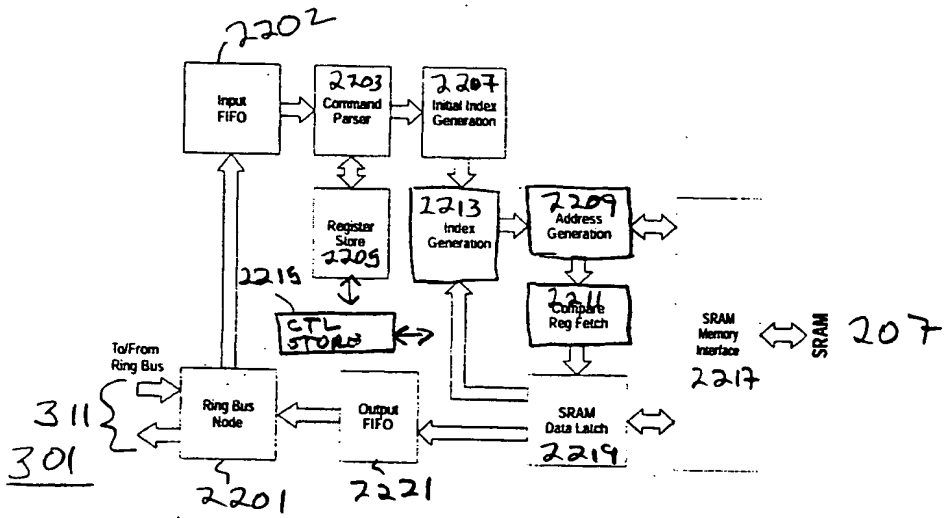

2202
Input FIFO
2203
Command Parser
2207
Initial Index Generation
Register Store 2205
2213
Index Generation
2209
Address Generation
2219
CTL STORE
2211
Compare Reg Fetch
SRAM Memory Interface
SRAM
207
2217
To/From Ring Bus
311
301
Ring Bus Node
Output FIFO
SRAM Data Latch
2219
2201
2221

Fig. 22

09/674864

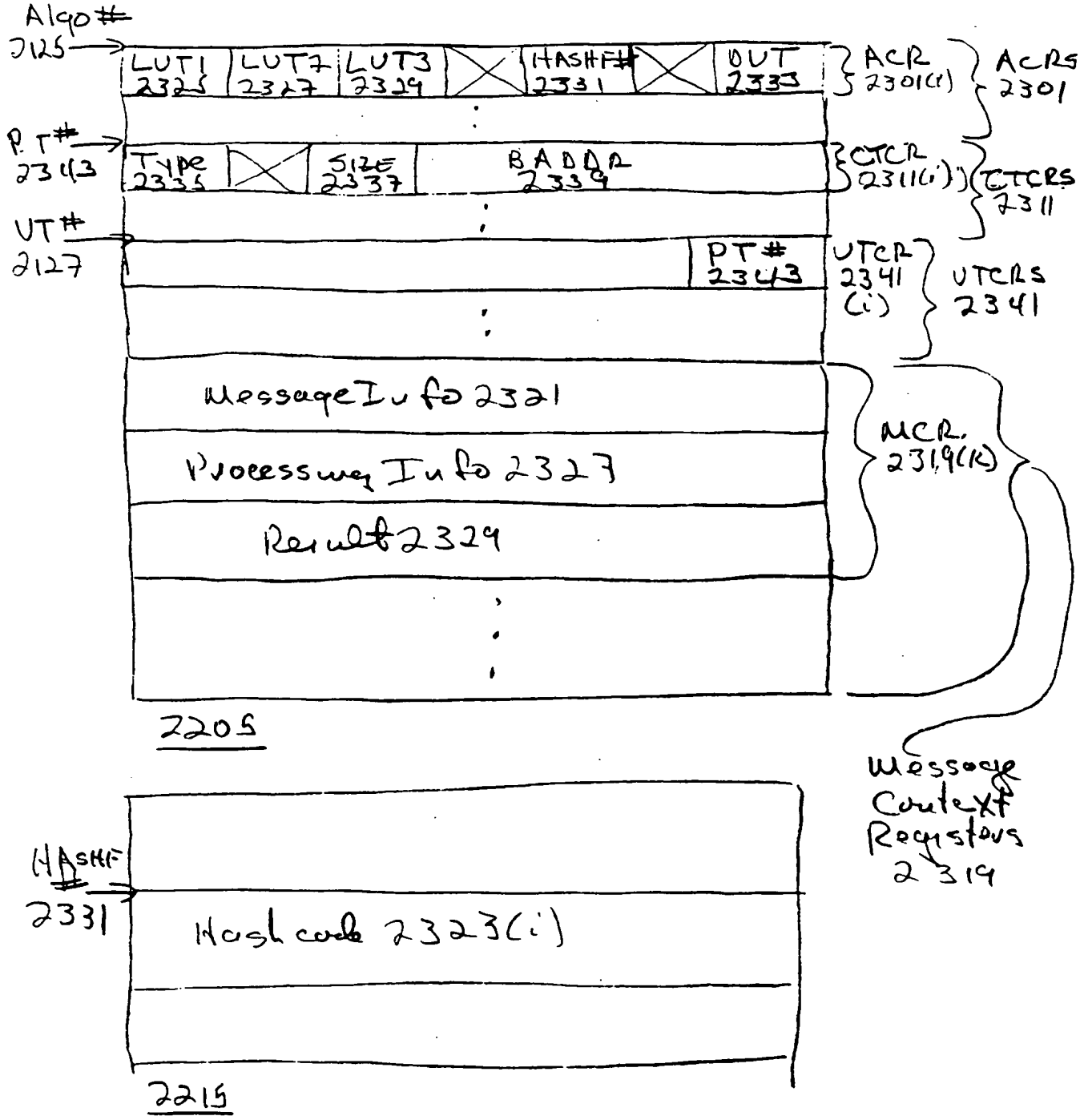


Fig. 23

	Command	Command ID	Return Data	Description	
2421	Write(vtable#, index, mask, data, offset, length)	0x2	None	Write data into a virtual table at index.	2401
	Read(vtable#, index, offset, length)	0x3	Data	Reads data from a virtual table.	2403
2423	Find(alg#, key)	0x6	Physical Table, Index, Error	Finds a *key* using *alg#*. Sets Ring Bus Error Flag if *key* is not found.	2405
	FindW(alg#, key, data, offset, length)	0x4	Pass/Fail, Index, Error	Writes *data* into a table using a *key*. Sets Ring Bus Error Flag if the key is not found.	2407
	FindR(alg#, key, data, offset, length)	0x5	Pass/Fail, Index, data	Reads *length* dwords of *data* from a *vtable#* using a *key* at *offset* dwords. Sets Ring Bus Error Flag if the key is not found.	2409
2425	XOR(vtable#, Index, data/pcrc, offset, mask, crc, last)	0x1	None or CRC in CRC mode.	XORs up to a 32 bit value to *offset*. Only masks of up to four consecutive bytes are valid. A special mode exists for CRC calculations.	2411
	Add(vtable#, index, data, offset, mask)	0x7	None	Adds up to a 32-bit value to *offset*. Only masks of up to four consecutive bytes are valid.	2413
2427	WriteReg(reg_addr, data)	0x0,0x10	None	Write *data* to TLE register at *reg_addr*.	2415
	ReadReg(reg_addr, data)	0x0,0x11	Data	Read *data* from TLE register at *reg_addr*.	2417
	Echo(data)	0x0,0x04	Data	Returns *data* from TLE. For test purposes.	2419
	Nop()	0x0,0x05	None	Inserts a NOP into the TLE pipe.	2420

Fig. 24

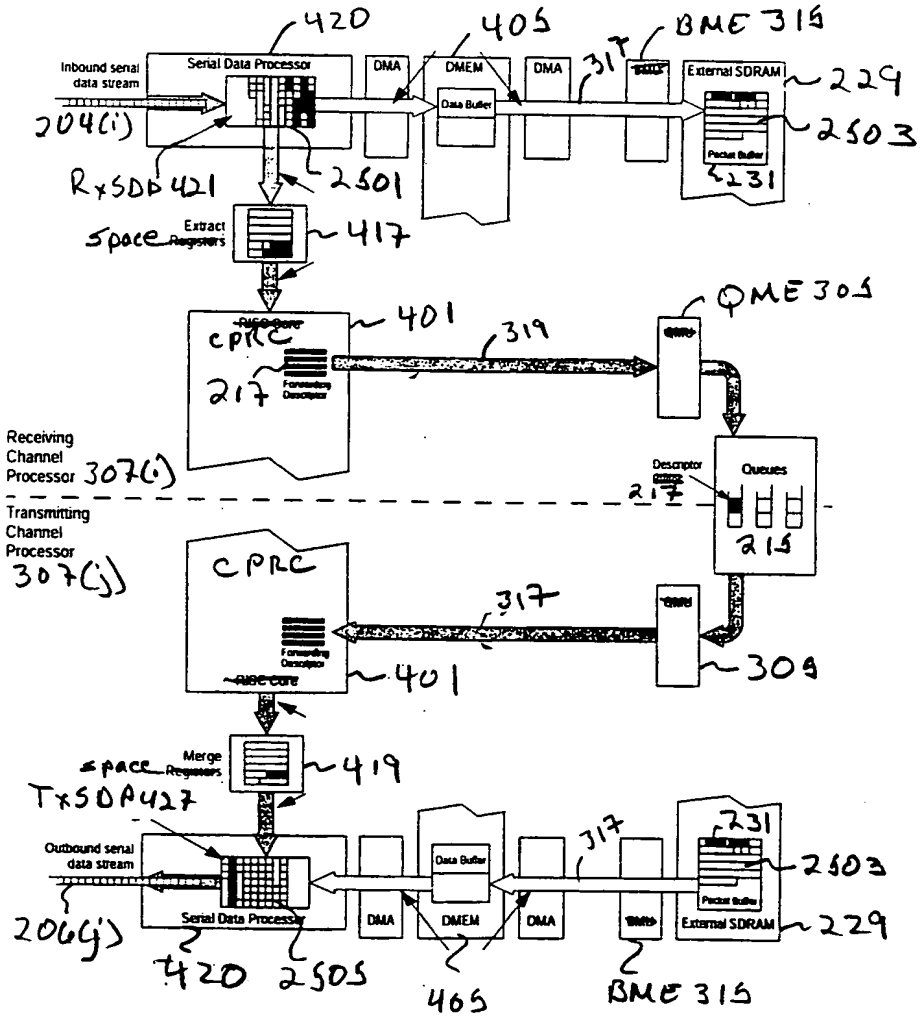

420
405
BME 315
Inbound serial data stream
Serial Data Processor
DMA
DMEM
DMA
317
External SDRAM
229
204(i)
Data Buffer
2503
Packet Buffer
231
RxSDP 421
2501
Space Extract Registers
417
401
CPRC
217
Forwarding Descriptor
319
QME 305
Receiving Channel Processor 307(i)
Transmitting Channel Processor 307(j)
Descriptor
217
Queues
215
CPRC
317
Forwarding Descriptor
AIBC Core
401
309
Space Merge Registers
419
TxSDP 427
Outbound serial data stream
Data Buffer
317
231
2503
Packet Buffer
External SDRAM
229
206(j)
Serial Data Processor
DMA
DMEM
DMA
420
2505
405
BME 315

Fig. 25

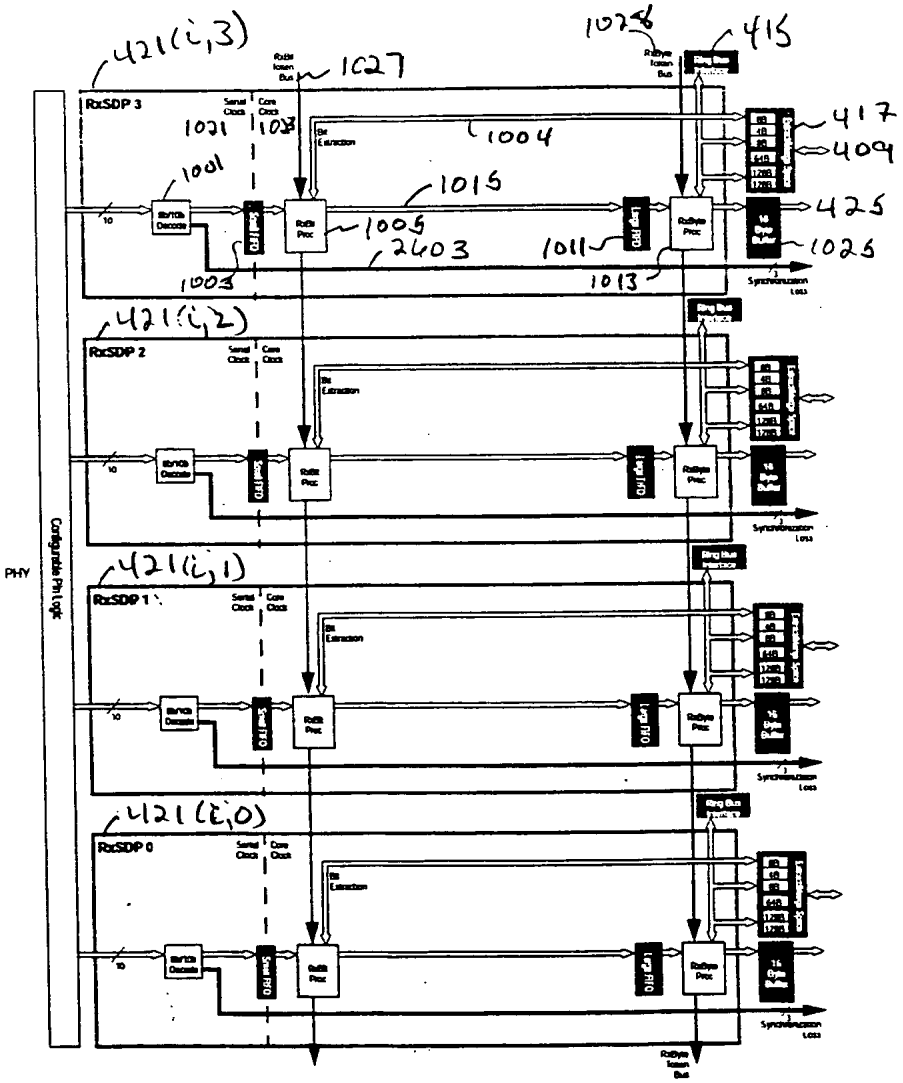

421(i,3)
1027
1027a
415
RxSDP 3
1021
1023
1004
417
409
1001
1015
425
1005
2603
1011
1025
1003
1013
421(i,2)
RxSDP 2
421(i,1)
RxSDP 1
421(i,0)
RxSDP 0
PHY
Configurable PHY Logic
2601

Fig. 26

09/674864

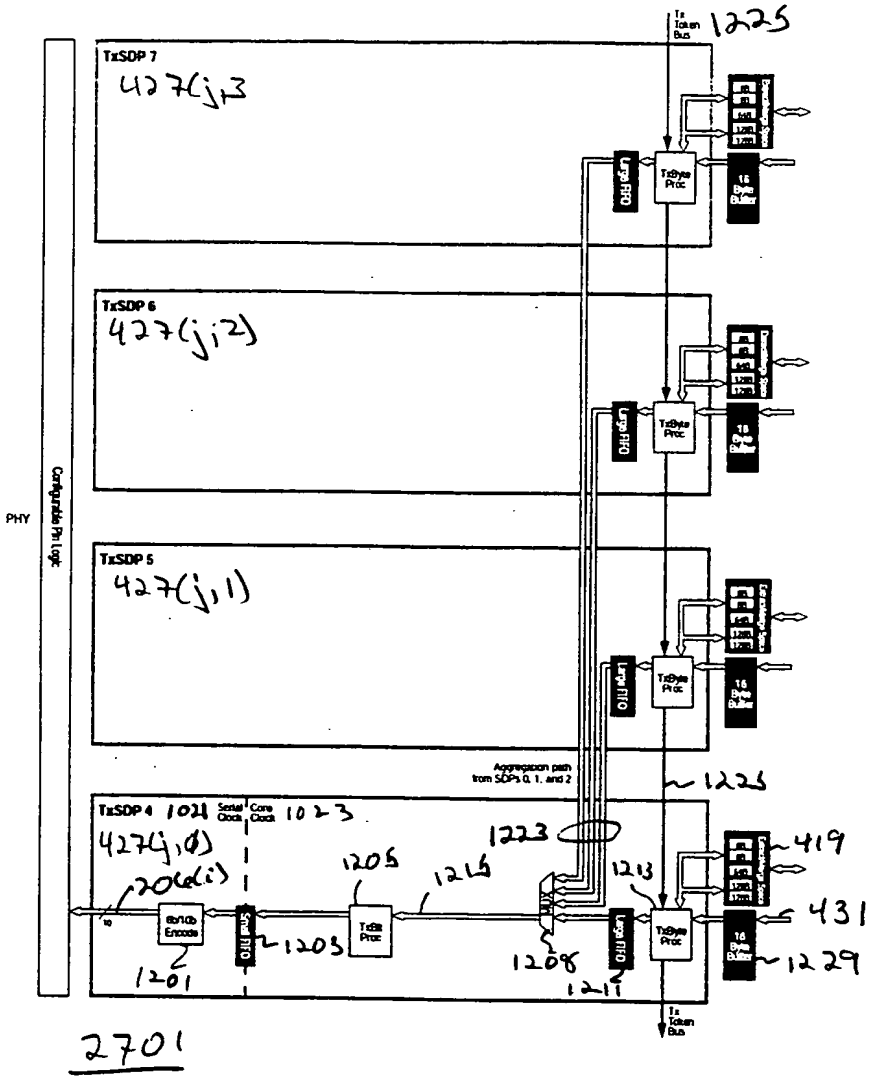
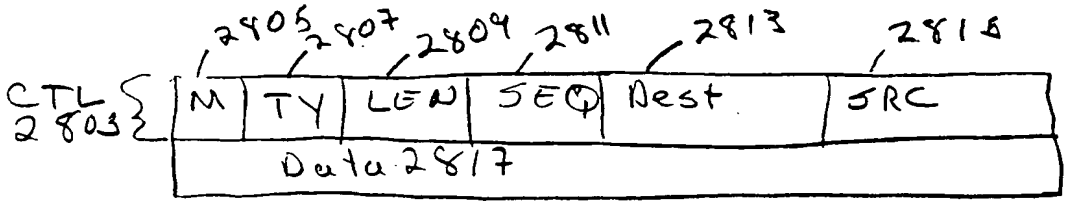


Fig. 27



Rambus message 2801

Fig. 28

09/674864

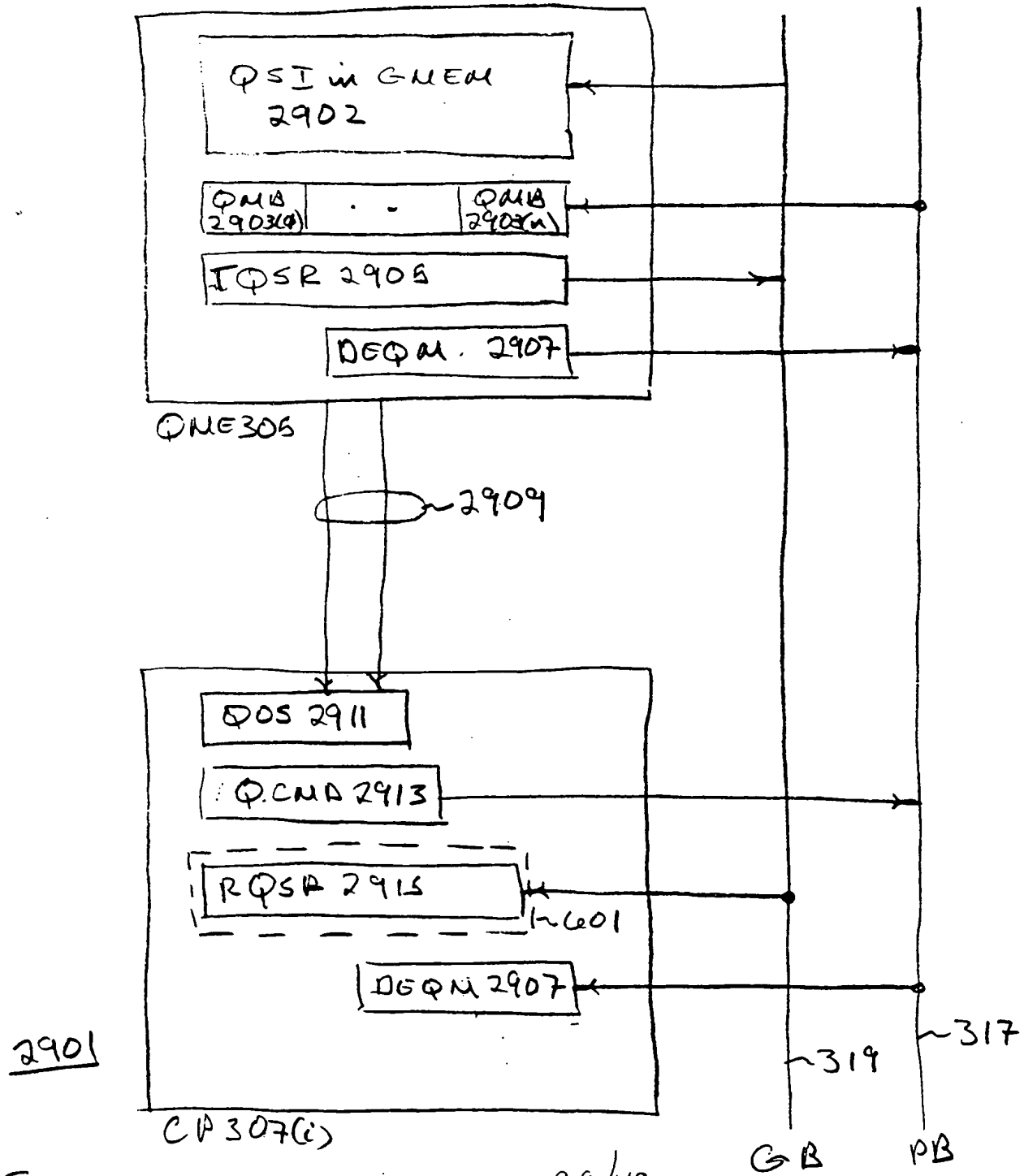
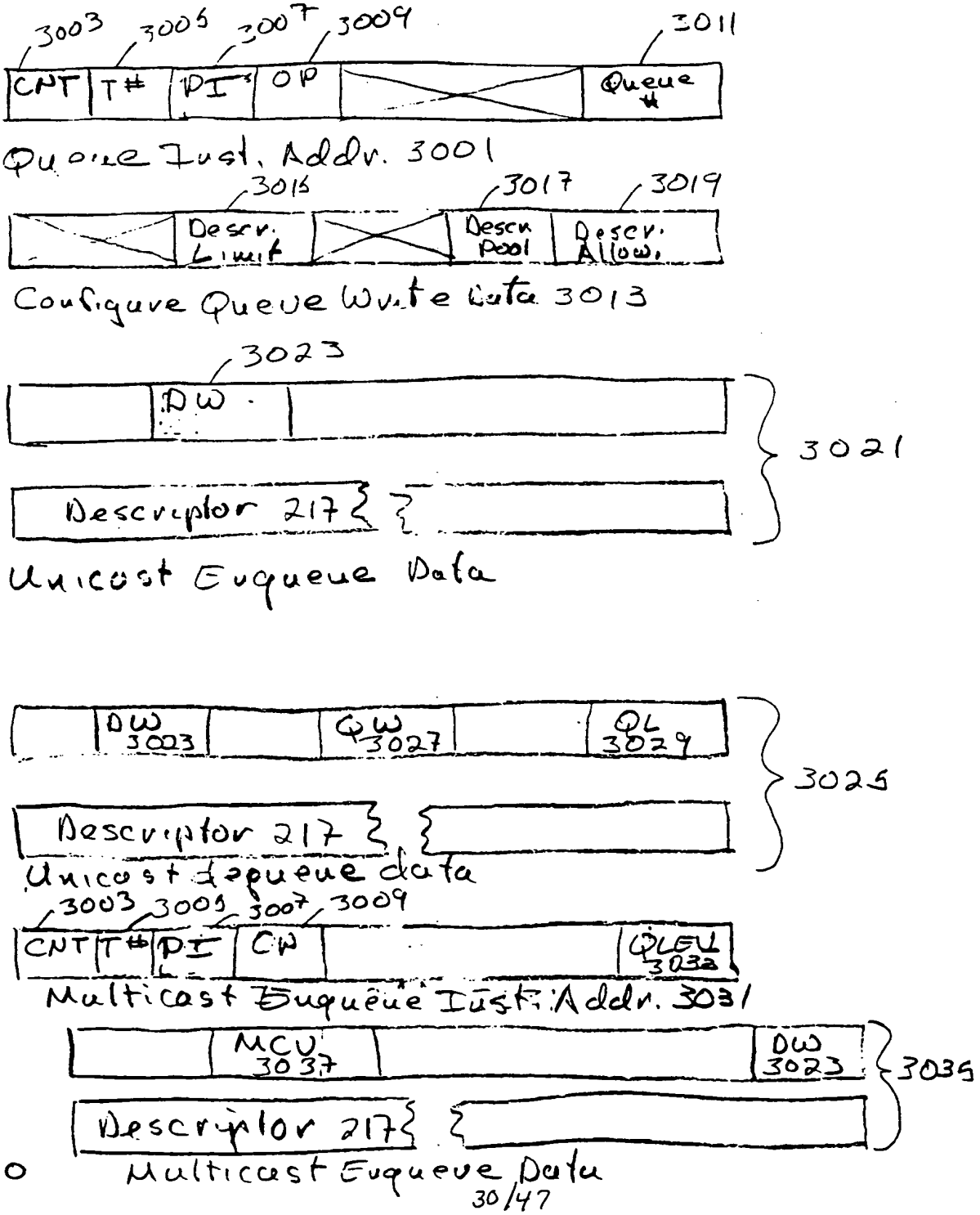


Fig. 29


3003
3005
3007
3009
3011
CNT
T#
PI
OP
Queue #
Queue Inst. Addr. 3001
3015
3017
3019
Descr. Limit
Descr Pool
Descr. Allow.
Configure Queue Write Data 3013
3023
DW
3021
Descriptor 217
Unicast Enqueue Data
DW 3003
QW 3027
QL 3029
3025
Descriptor 217
Unicast Enqueue data
3003
3005
3007
3009
CNT
T#
PI
CW
QLEN 3033
Multicast Enqueue Inst. Addr. 3031
MCU 3037
DW 3023
3035
Descriptor 217
Multicast Enqueue Data

2913

Fig. 30

09/674864

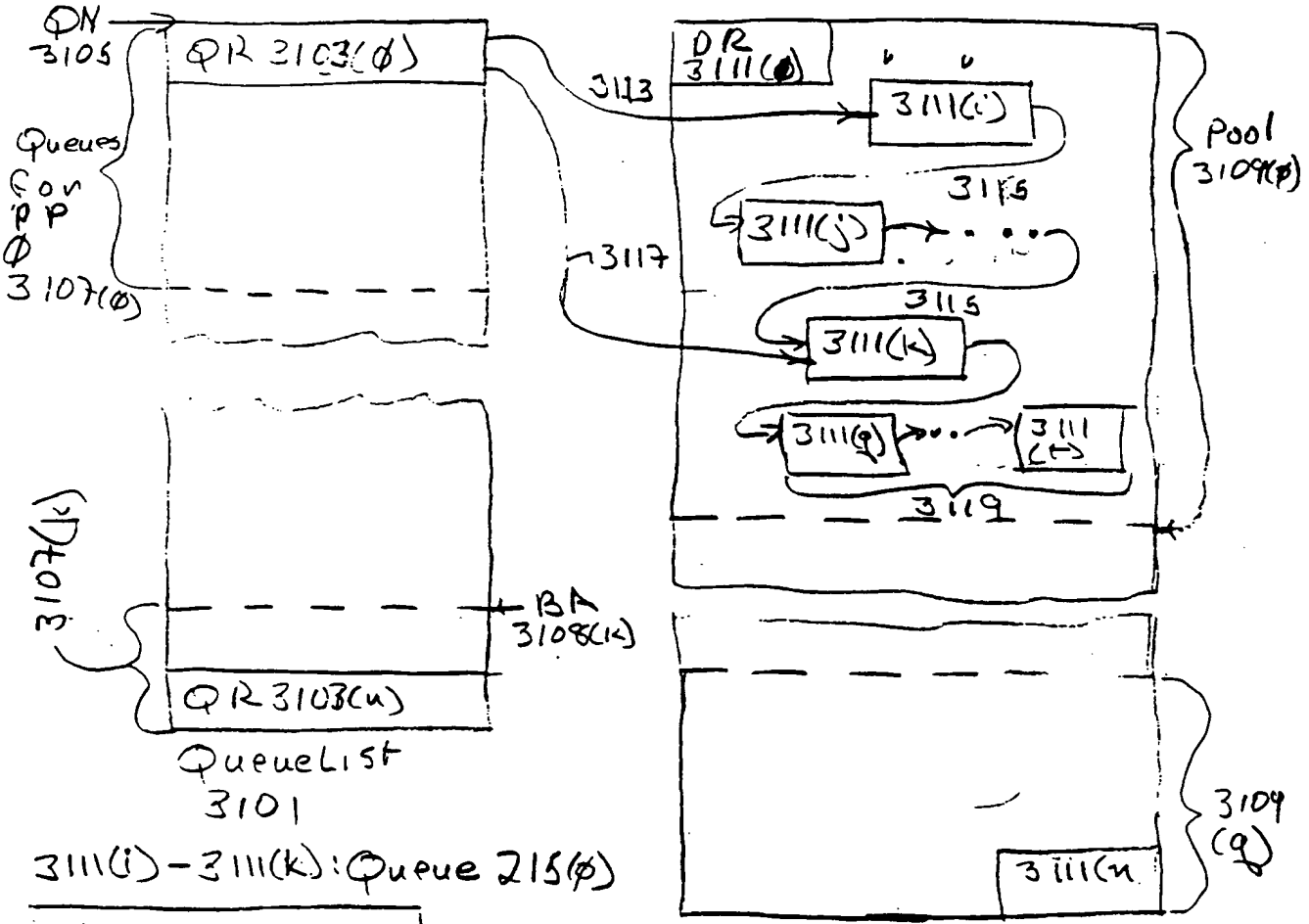


Fig. 31

09/674864

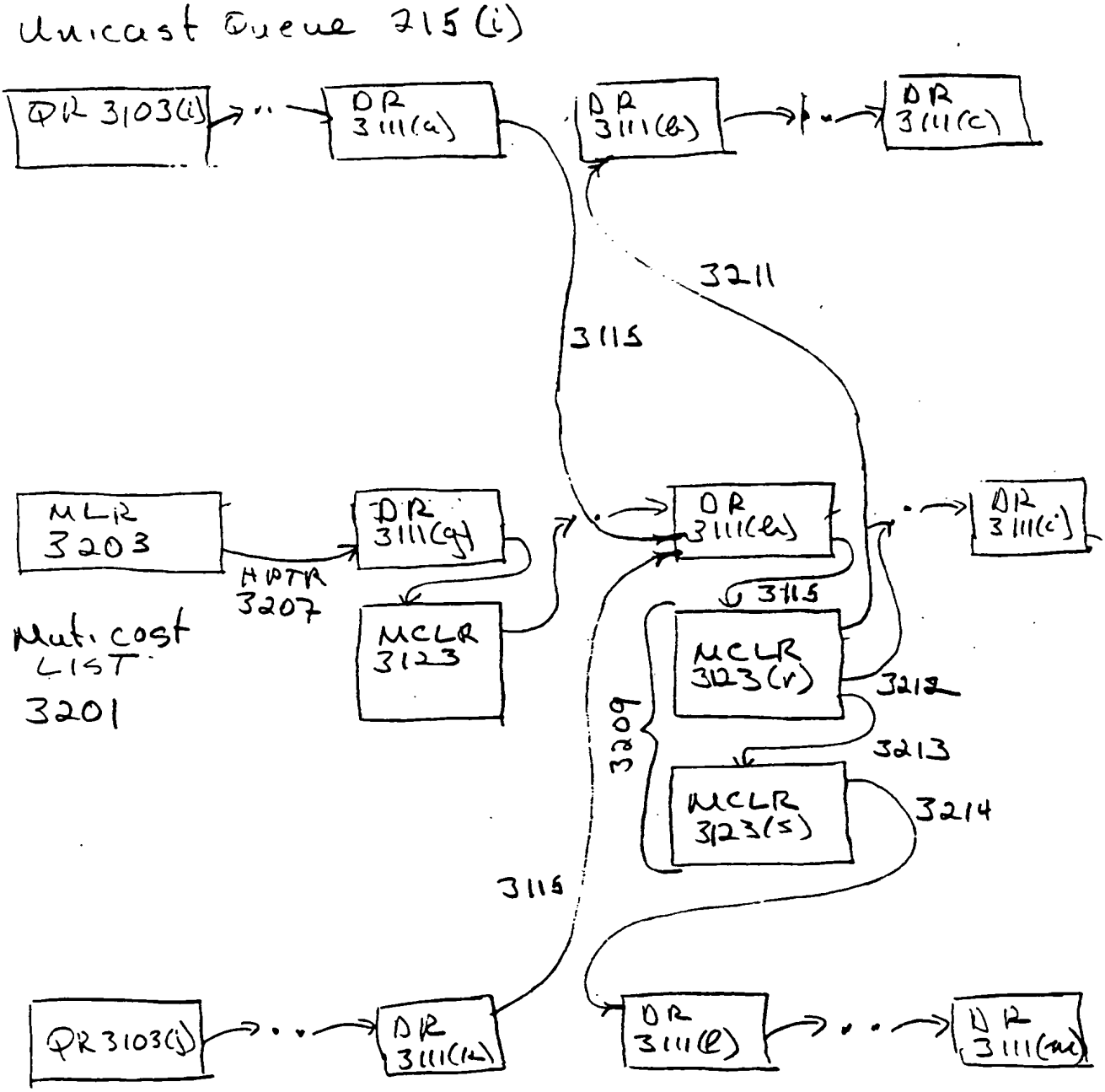


Fig. 32

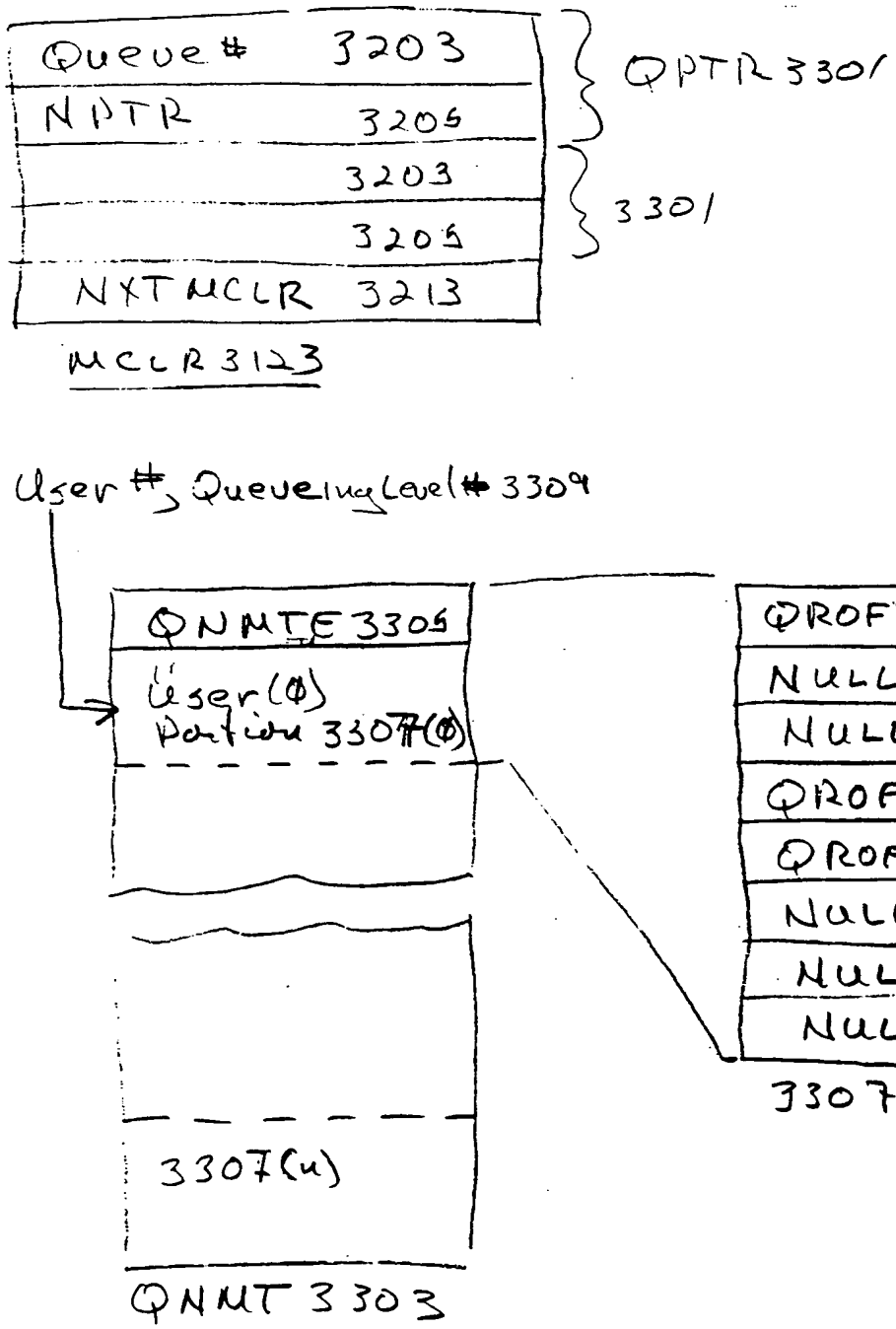

Queue # 3203
NPTR 3205
3203
3205
NXT MCLR 3213
QPTR 3301
3301
MCLR 3123
User #, Queuing Level # 3309
3305
QNMTE 3305
User (0) Portion 3307(0)
3307(u)
QNMT 3303
QROFF NUM (0,0)
NULL (0,1)
NULL (0,2)
QROFF NUM (0,3)
QROFF NUM (0,4)
NULL (0,5)
NULL (0,6)
NULL (0,7)
3307(0)

Fig. 33

09/674864

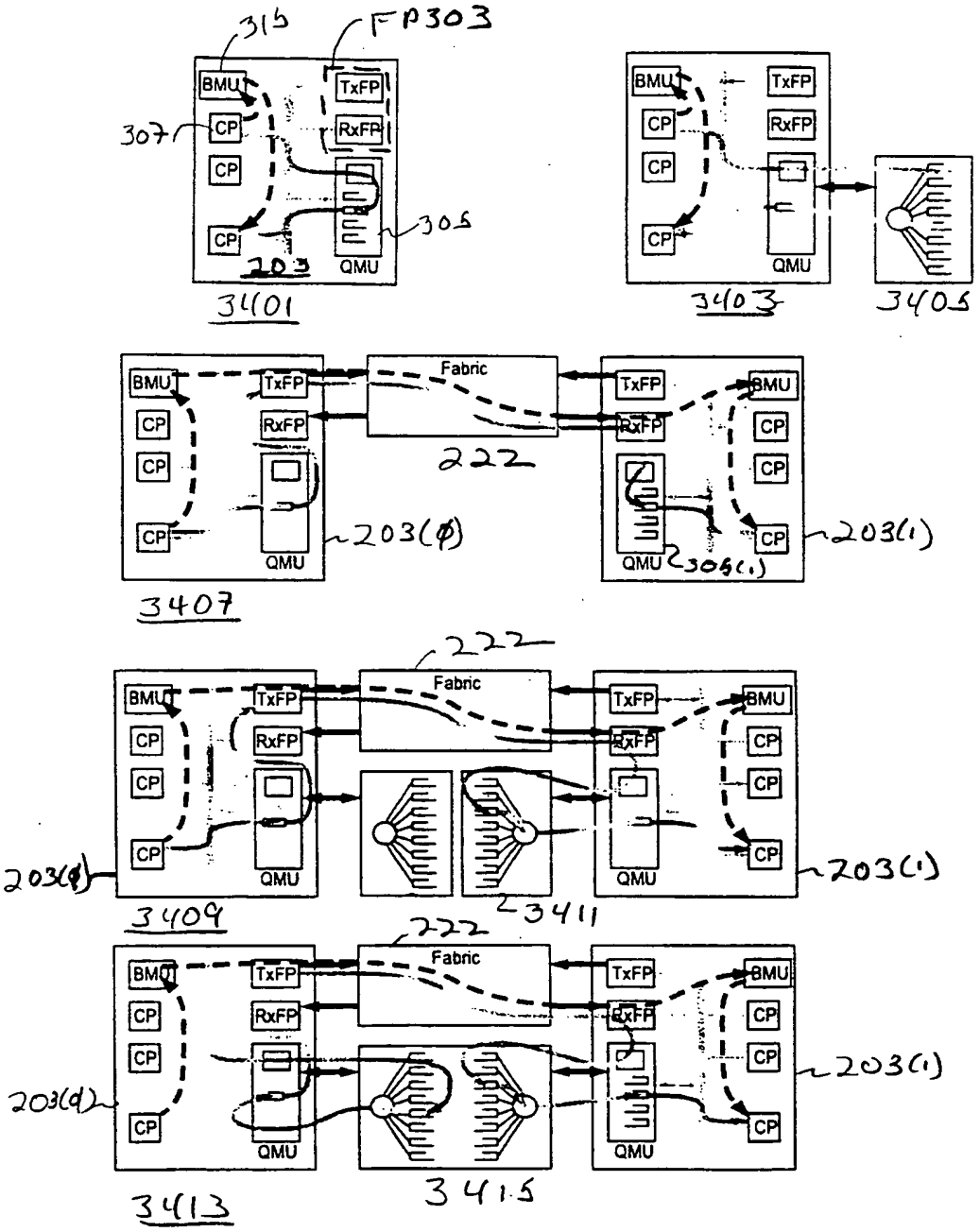
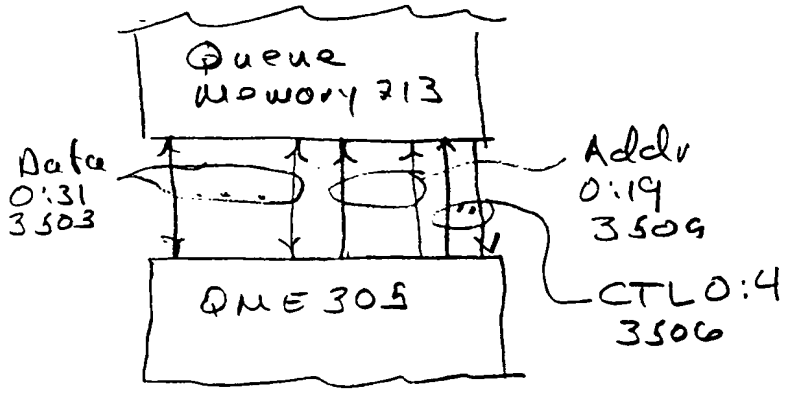
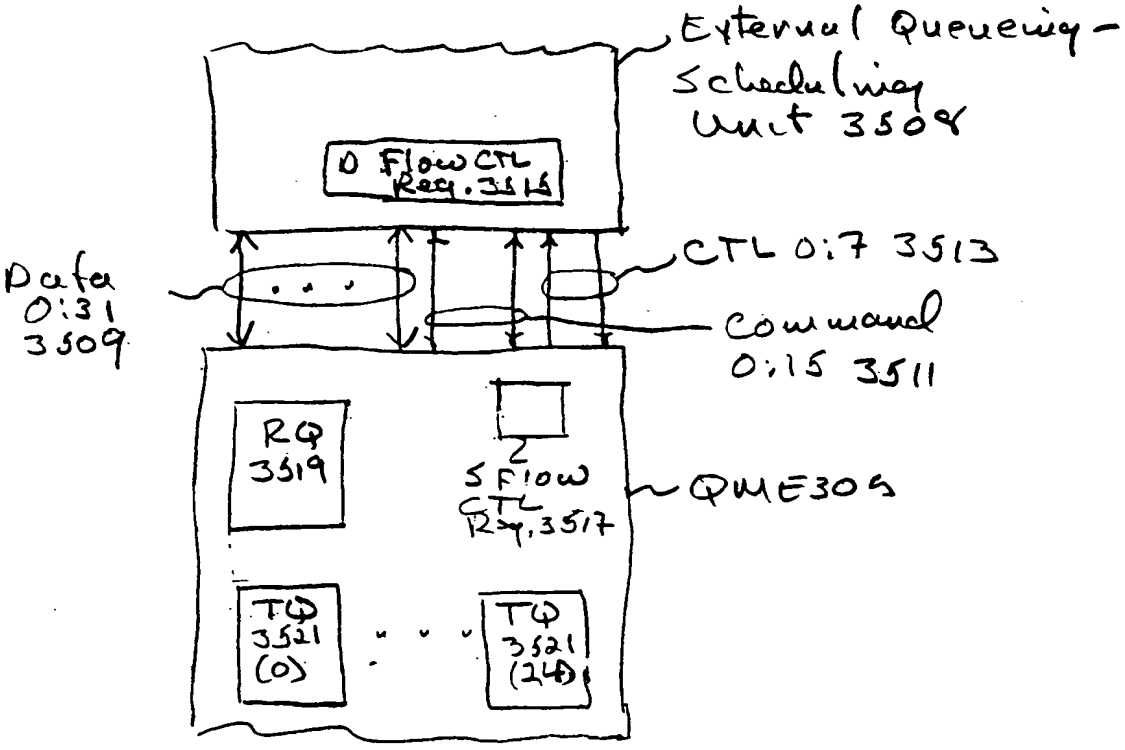


Fig. 34



Memory External Interface 3501



Scheduler External Interface 3507

Fig. 35

09/674864

Pins 3601 Direction 3603

3605	Clk	1	DCP	->	SCHED	
3607	D_Flow_Ctrl	3	DCP	->	SCHED	
3609	S_Flow_Ctrl	1	DCP	<-	SCHED	; If = 0, the Scheduler can ; accept at least one descriptor.
3611	Xfer_Rqst	1	DCP	<-	SCHED	; If = 1, the Scheduler has at ; least one descriptor to transfer.
3613	Xfer_Ctrl	2	DCP	->	SCHED	
3511	Command Data	16	DCP	<->	SCHED	
	Cmd_Parity	1	DCP	<->	SCHED	
3509	Descript Data	32	DCP	<->	SCHED	
	Data_Parity	1	DCP	<->	SCHED	
	Total	58				

513

3507

First Command Code 3615

1 1 1 1 1 1
5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0

Destination DCP Processor Number 3617

3514

Fig. 36

09/674864

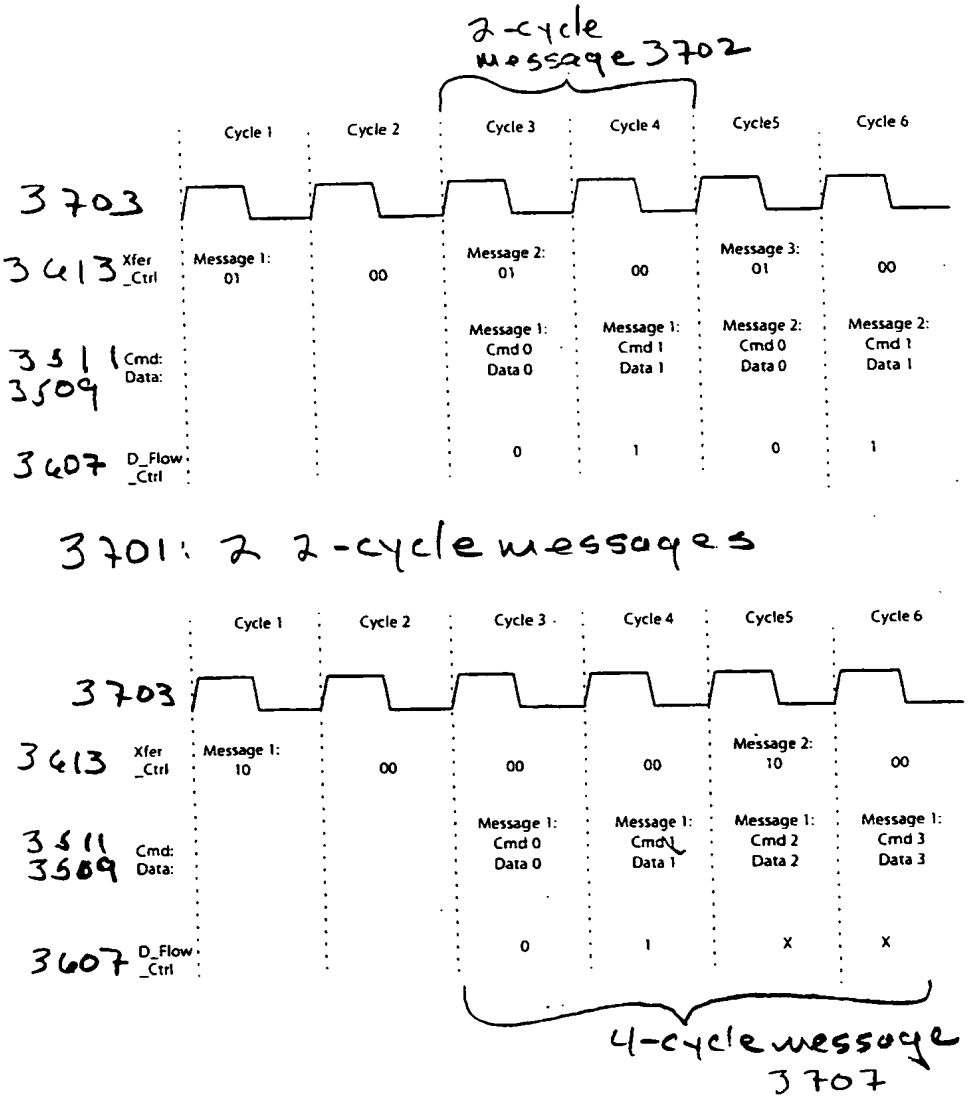


Fig. 37

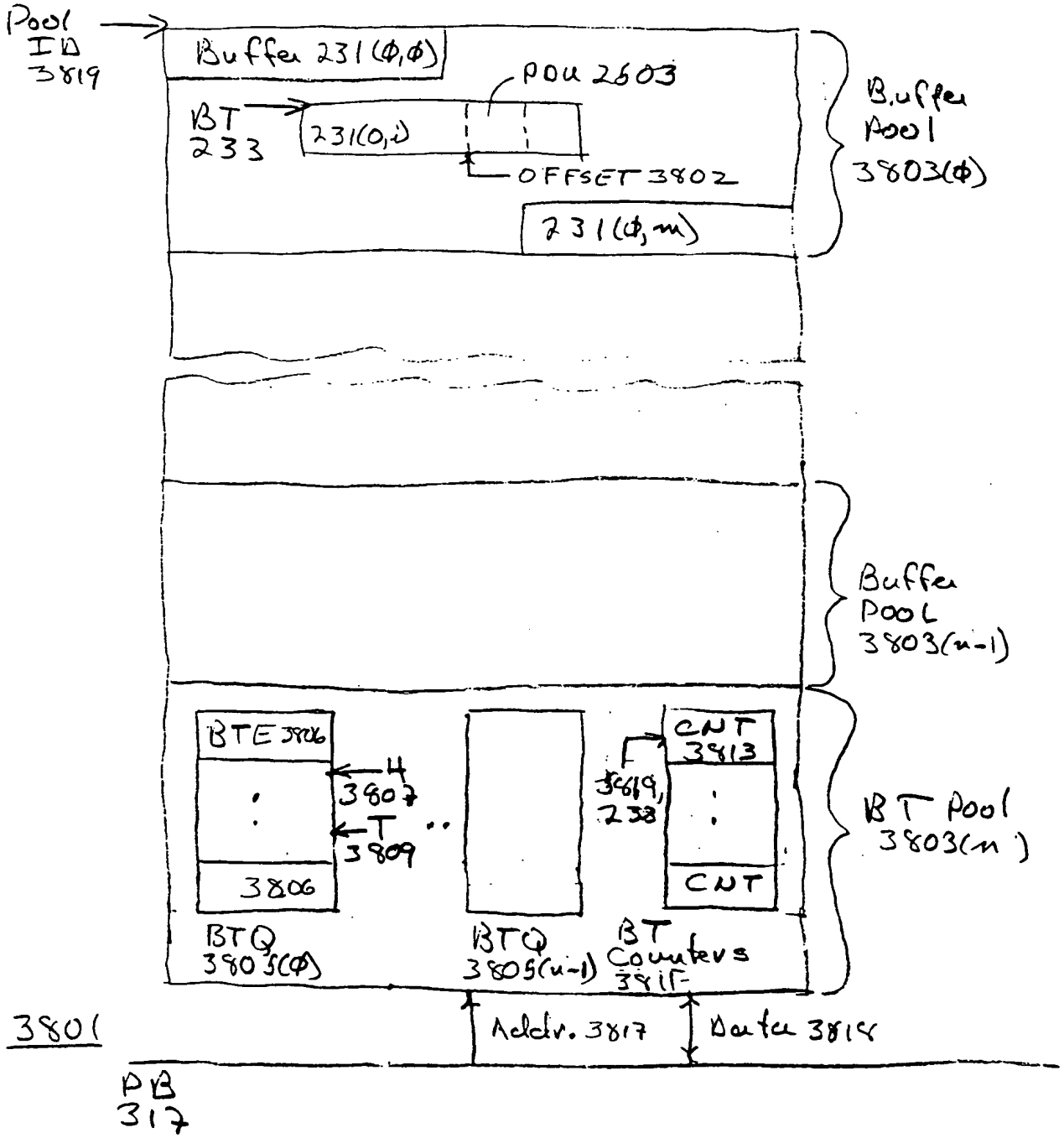
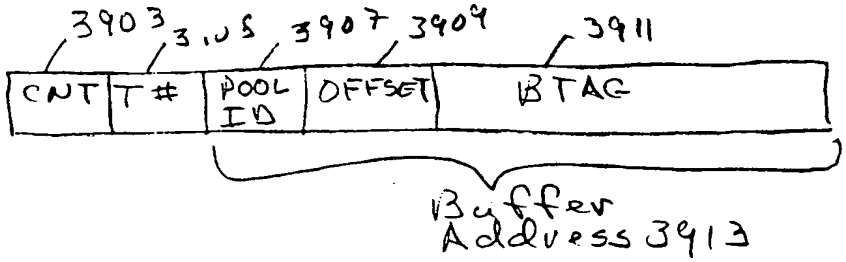
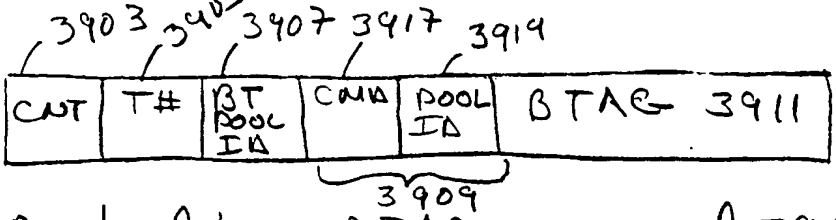

Pool ID 3819
Buffer 231(0,0)
POU 2603
BT 233
231(0,i)
OFFSET 3802
231(0,m)
Buffer Pool 3803(0)
Buffer Pool 3803(n-1)
BTE 3806
H 3807
T 3809
3806
CNT 3813
3819, 238
CNT
BT Pool 3803(n)
BTQ 3805(0)
BTQ 3805(n-1)
BT Counters 3811
3801
Addr. 3817
Data 3815
PB 317

Fig. 38

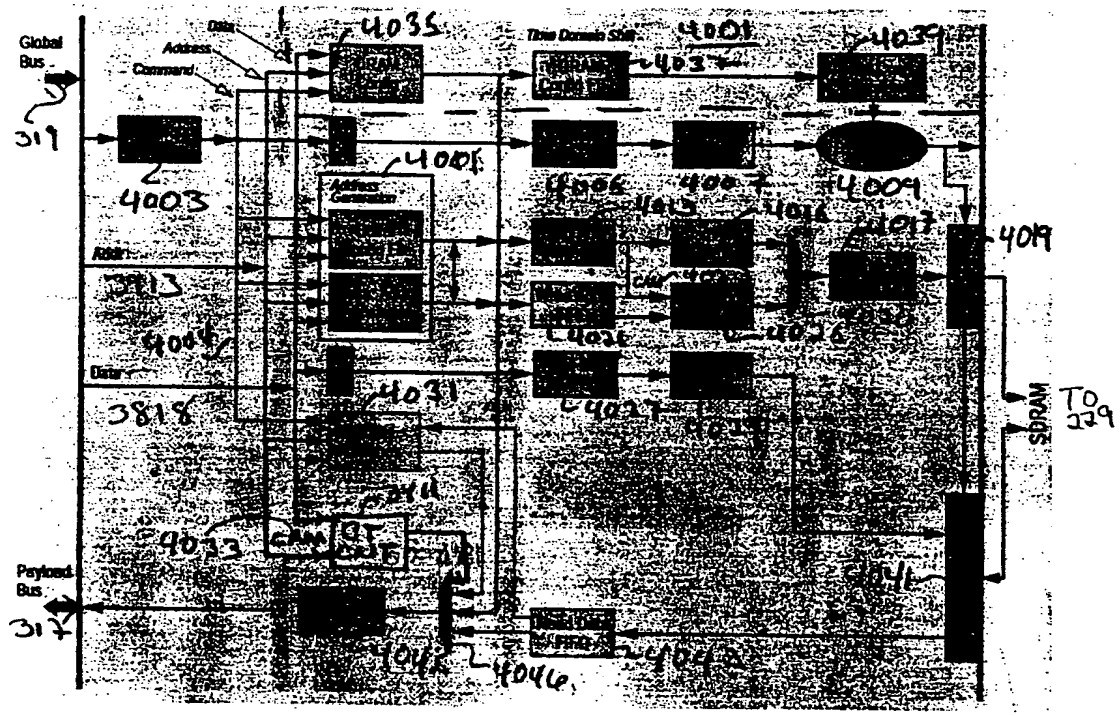


Payload bus
Buffer read/write command 3901



Payload bus BTAG command 3915

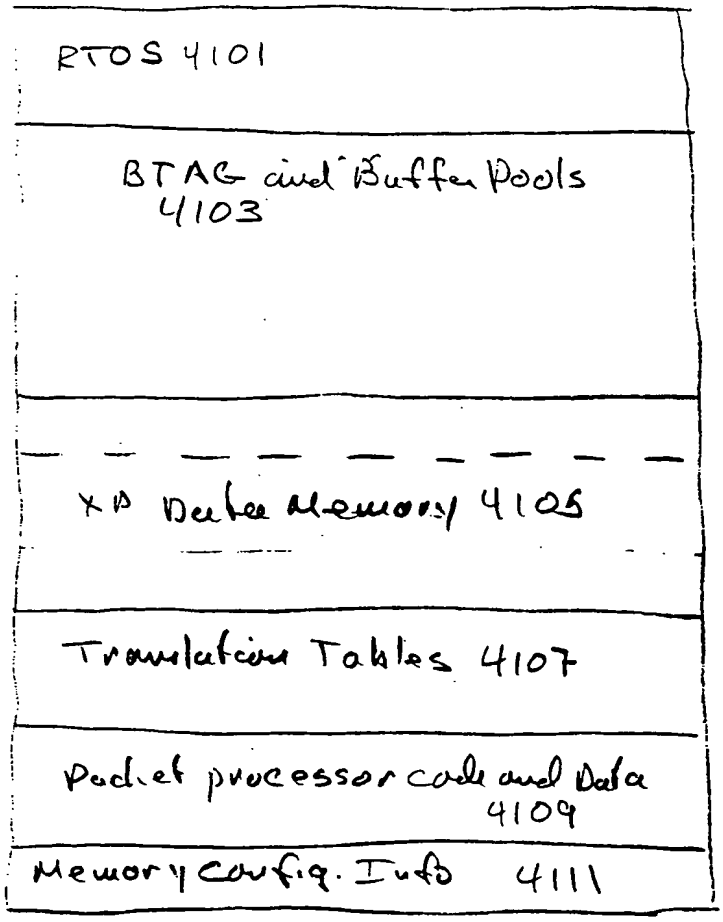
Fig. 39


Global Bus
Address
Command
Data
Payload Bus
SDRAM
TO 329
4035
4001
4039
4003
4009
4005
4007
4011
4013
4015
4017
4019
4021
4023
4025
4027
4029
4031
4033
4041
4043
4045
4046
317
313
3818

316

Fig. 4D

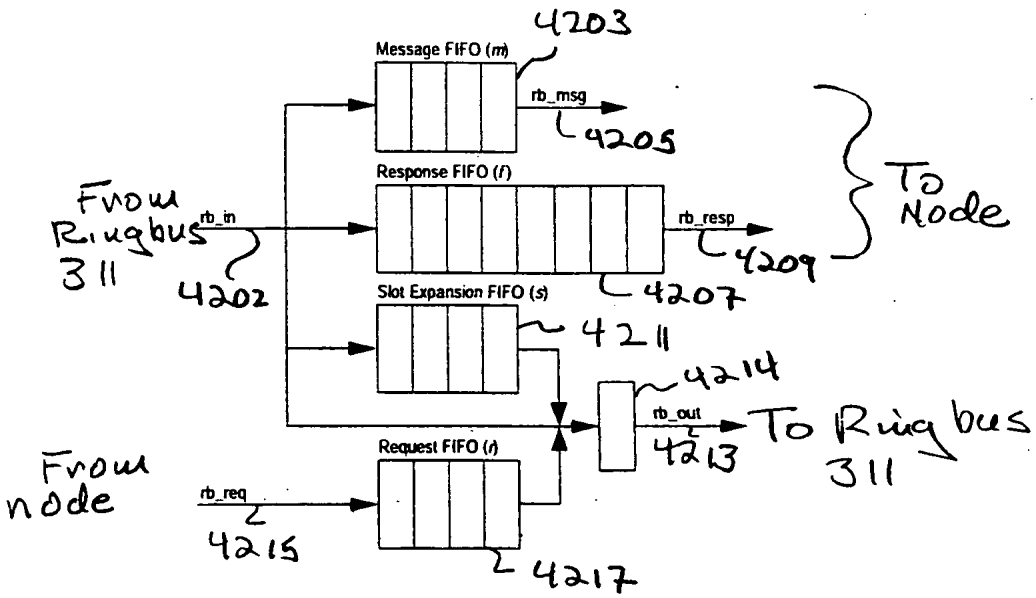
09/674804



229

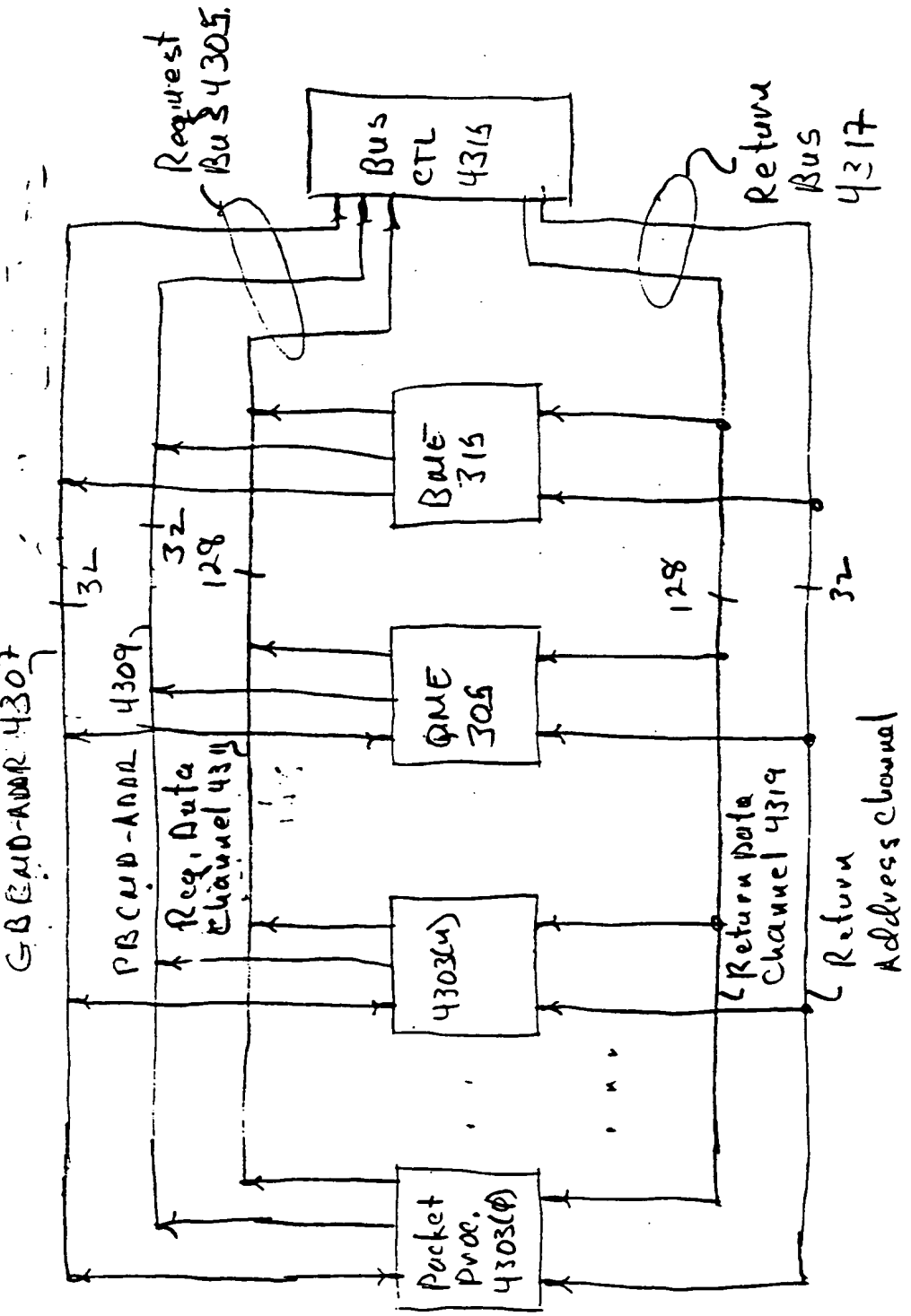
Fig. 41

09/674864



Ringbus Node Interface 4201

Fig. 42


Request Bus 4305
Bus CTL 4315
Return Bus 4317
BME 315
QME 305
4303(1)
Packet Proc. 4303(Φ)
GBEAD-ADDR 4307
PBCMD-ADDR
4309
Req. Data Channel 4311
Return Data Channel 4319
Return Address channel 4321
32
128
32
128
32

4301

Fig. 43

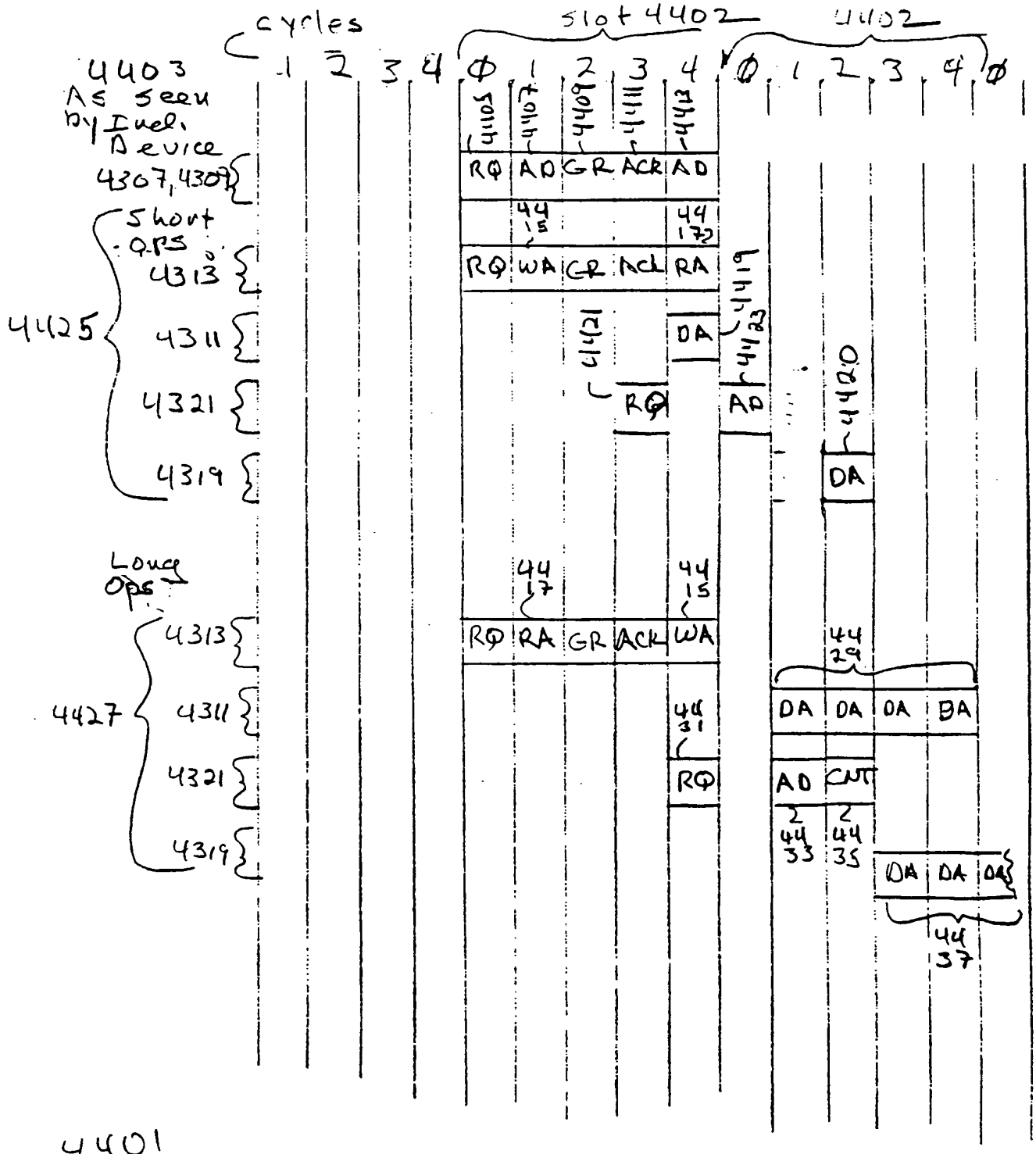

cycles
slot 4402
4402
4403
As seen by Indiv. Device
4307, 4309
short ops
4313
4311
4321
4319
4425
long ops
4313
4311
4321
4319
4427
RQ AD GR ACK AD
RQ WA GR ACK RA
DA
RQ
RQ RA GR ACK WA
DA DA DA DA
RQ
AD CNT
DA DA DA
4401

Fig. 44

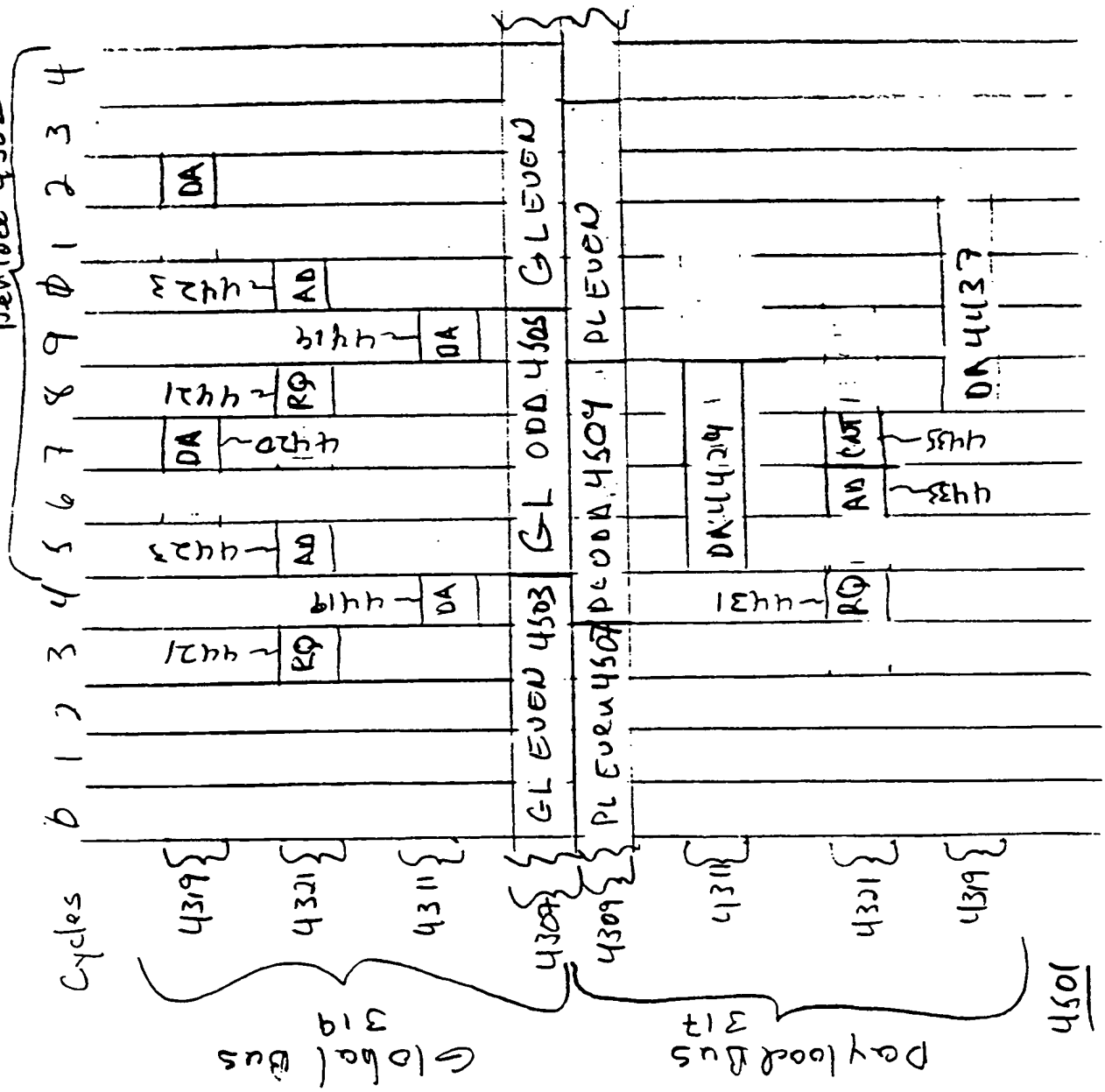

Period 4502
0 1 2 3 4 5 6 7 8 9 Φ 1 2 3 4
Cycles
4319
4321
4311
4307
4309
4313
4321
4339
Global Bus 312
Poy loop Bus 312
4501
GL EVEN 4503
GL ODD 4505
GLEVEN
PL EVEN 4504
PL ODD 4506
PLEVEN
DA 4437
DA 4434
DA
RQ
AD
CAS

Fig. 45

4603 4605 4607 4609 4611 4613 4615 4617 4619

Pin	Purpose	RMII	OC-3	DS1	DS3	GMII (Tx)	GMII (Rx)	TBI (Tx)	TBI (Rx)	OC-12
CP0_0	outclk	REF_CLK	RCLK_H	TCLK	TCLK	TCLK	nc	TCLK	nc	TCLK
_1	inclk	CRS_DV	RCLK_L	RCLK	RCLK	CRS	nc		nc	TCLK1
_2	data	TXD[0]	TXD_H	TDATA	TDATA	TXD[0]	nc	TXD[0]	nc	TXD[0]
_3	data	TXD[1]	TXD_L	TxFrame	TxFrame	TXD[1]	nc	TXD[1]	nc	TXD[1]
_4	data	RXD[0]	RXD_H	RDATA	RDATA	TXD[2]	nc	TXD[2]	nc	TXD[2]
_5	data	RXD[1]	RXD_L	RxFrame	RxFrame	TXD[3]	nc	TXD[3]	nc	TXD[3]
_6	data	TX_EN	SIGNAL_DET			TX_EN	nc	TXD[8]	nc	
CP1_0	outclk	REF_CLK	RCLK_H	TCLK	TCLK					
_1	inclk	CRS_DV	RCLK_L	RCLK	RCLK	COL	nc			
_2	data	TXD[0]	TXD_H	TDATA	TDATA	TXD[4]	nc	TXD[4]	nc	TXD[4]
_3	data	TXD[1]	TXD_L	TxFrame	TxFrame	TXD[5]	nc	TXD[5]	nc	TXD[5]
_4	data	RXD[0]	RXD_H	RDATA	RDATA	TXD[6]	nc	TXD[6]	nc	TXD[6]
_5	data	RXD[1]	RXD_L	RxFrame	RxFrame	TXD[7]	nc	TXD[7]	nc	TXD[7]
6	data	TX_EN	SIGNAL_DET			TX_ER	nc	TXD[9]	nc	
CP2_0	outclk	REF_CLK	RCLK_H	TCLK	TCLK					
_1	inclk	CRS_DV	RCLK_L	RCLK	RCLK	nc	RCLK	nc	RCLK	RCLK
_2	data	TXD[0]	TXD_H	TDATA	TDATA	nc	RXD[0]	nc	RXD[0]	RXD[0]
_3	data	TXD[1]	TXD_L	TxFrame	TxFrame	nc	RXD[1]	nc	RXD[1]	RXD[1]
_4	data	RXD[0]	RXD_H	RDATA	RDATA	nc	RXD[2]	nc	RXD[2]	RXD[2]
_5	data	RXD[1]	RXD_L	RxFrame	RxFrame	nc	RXD[3]	nc	RXD[3]	RXD[3]
_6	data	TX_EN	SIGNAL_DET			nc	RX_DV		RXD[8]	FP
CP3_0	outclk	REF_CLK	RCLK_H	TCLK	TCLK					
_1	inclk	CRS_DV	RCLK_L	RCLK	RCLK			nc	RCLKN	
_2	data	TXD[0]	TXD_H	TDATA	TDATA	nc	RXD[4]	nc	RXD[4]	RXD[4]
_3	data	TXD[1]	TXD_L	TxFrame	TxFrame	nc	RXD[5]	nc	RXD[5]	RXD[5]
_4	data	RXD[0]	RXD_H	RDATA	RDATA	nc	RXD[6]	nc	RXD[6]	RXD[6]
_5	data	RXD[1]	RXD_L	RxFrame	RxFrame	nc	RXD[7]	nc	RXD[7]	RXD[7]
_6	data	TX_EN	SIGNAL_DET			nc	RX_ER	nc	RXD[9]	LOCKDET

Fig. 46 4601

09/674864

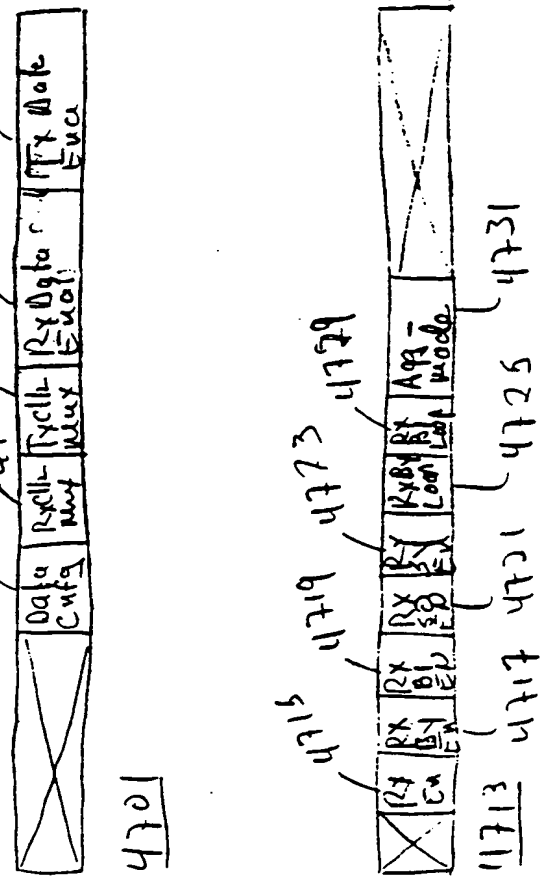


Fig. 47